ACCESS - July 2001
Twig example with test procedure

In order to make the twig from yesterday's notes, as well as more complicated examples, it is nice to have a test procedure to make sure you have picked your affine maps correctly (and to help you adjust them later if necessary.) The procedure TESTMAP below, takes an affine function as its input, and the result is a mapping -L picture like the ones in the fractal templates.

```
> restart:
```

```
> with(plots):
  Digits:=4:
Warning, the name changecoords has been redefined
```

```
> TESTMAP:=proc(f)   #this procedure lets you test individual
                     #functions in your iterated function system
    local S,  #corners of unit square
          L,  #corners of letter L
          Sq, #unit square
          Llet,  #letter L
          AS, #transf of square corners
          ASq,#transf of square
          AL, #transf of L corners
          ALlet;  #transf of letter L
  S:=[[0,0],[0,1],[1,1] ,[1,0]];
  L:=[[.1,.9],[.1,.75],[.2,.75],[.2,.775],[.125,.775],[.125,.9]]:
  Sq:=polygonplot(S): #polygonplot connects the dots!
  Llet:=polygonplot(L):
  AS:=map(f,S):
  AL:=map(f,L):
  ASq:=polygonplot(AS):
  ALlet:=polygonplot(AL):
     #finally, display the unit square, the letter L,
     #and how they are transformed by f:
  display({Sq,Llet,ASq,ALlet},scaling=constrained,
  title=`test picture`);
  end:
```

Here is the standard affine map, which encodes

$$\text{AFFINE1}\left(\begin{bmatrix} x \\ y \end{bmatrix}\right)=\begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}+\begin{bmatrix} e \\ f \end{bmatrix}$$

```
> AFFINE1:=proc(X,a,b,c,d,e,f)
  RETURN(evalf([a*X[1]+b*X[2]+e,
                c*X[1]+d*X[2]+f]));
  end:
```

And in case you want to use it, an alternative version called AFFINE2 which lets you specify scaling factors and roation angles instead;

$$\mathrm{AFFINE2}\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} r\cos(\alpha) & -s\sin(\beta) \\ r\sin(\alpha) & s\cos(\beta) \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

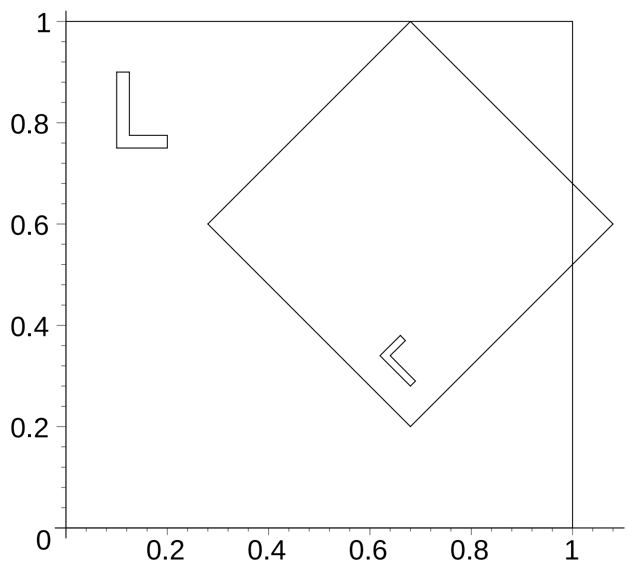
```
> AFFINE2:=proc(X,r,alpha,s,beta,e,f)
  RETURN(AFFINE1(X,r*cos(alpha),-s*sin(beta),
           r*sin(alpha),s*cos(beta),e,f));
  end:
```

In the following example I tried to reproduce the mapping templates which gave the twig shown in Monday's notes, which came from Peitgen's book. It took several tries to get it approximately right, and then a number of readjustments to make the branches match up correctly. What you see are the final parameter values which were chosen.

```
> f1:=P->AFFINE1(P,.4,.4,.4,-.4,.28,.6);
```

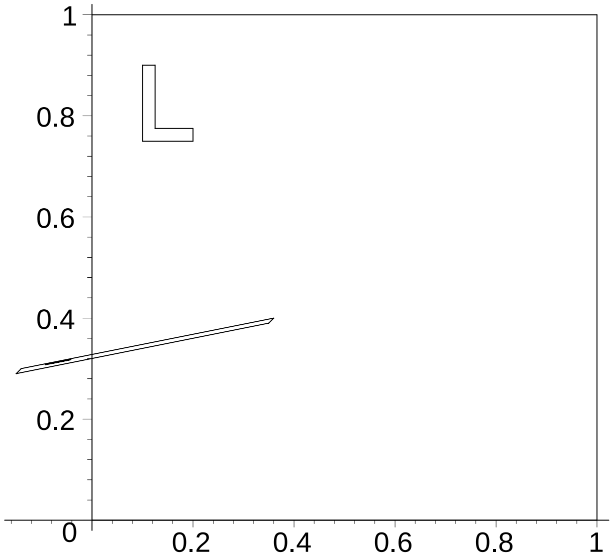
$$f1 := P \to \mathrm{AFFINE1}(P, .4, .4, .4, -.4, .28, .6)$$

```
> TESTMAP(f1);
```



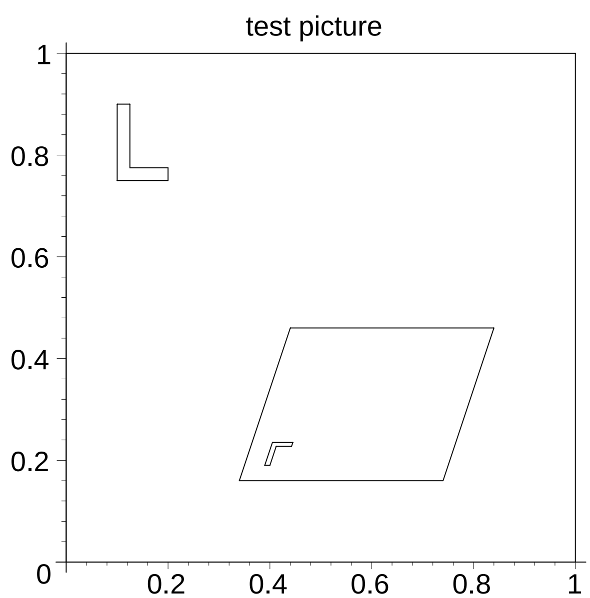

```
> f2:=P->AFFINE1(P,.5,.01 ,.1,.01 ,-.15,.29);
  TESTMAP(f2);
```

$$f2 := P \to \mathrm{AFFINE1}(P, .5, .01, .1, .01, -.15, .29)$$




```
> f3:=P->AFFINE1(P,.4,-.1 ,0 ,-.3 ,.44,.46);
  TESTMAP(f3);
```

$$f3 := P \rightarrow \mathrm{AFFINE1}(P, .4, -.1, 0, -.3, .44, .46)$$

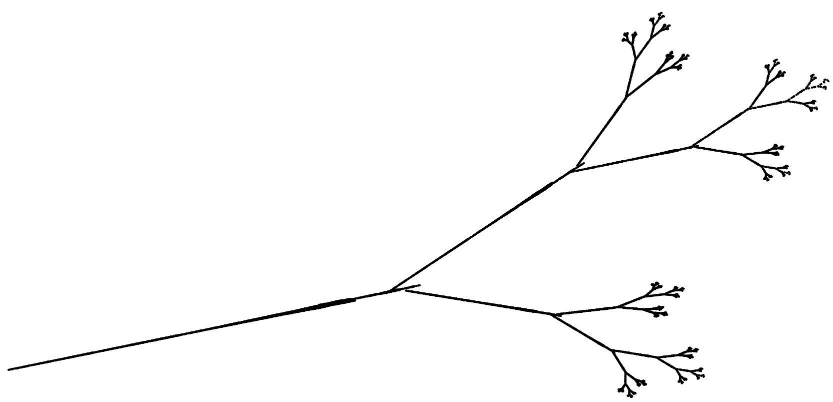



```
> S:={[0,0]};
```

$$S := \{[0, 0]\}$$

```
> for i from 1 to 9 do
  S1:=map(f1,S):
  S2:=map(f2,S);
```

```
   S3:=map(f3,S);
   S:=`union`(S1,S2,S3);
   od:
>
> pointplot(S,scaling=constrained,symbol=point,
   title=`twig`);
```



```
>
```