

Testing an affine transformation to see if it's a Euclidean contraction

If you define a function using AFFINE1 (from the Lpictures file), you may want to make sure it's a contraction to make sure your IFS will generate a fractal. The following procedure will do this: First, make sure you load the linear algebra library so that the procedure can use matrix commands: This procedure depends on a linear algebra theorem known as the singular value decomposition of a matrix, see Math 2270.

```
> with(linalg): #linear algebra library
Warning, the protected names norm and trace have been redefined and unprotected

> STRETCHTEST:=proc(fcn) #test affine map for contraction

    local Atemp, #matrix of affine transformation
        temp; #hold eigenvector information of the transpose
              #of Atemp times Atemp - the square root of
              #the largest eigenvalue
              #is the largest stretch factor -
              #you want this to be less than one

    Atemp:=transpose
        (matrix([fcn([1.,0])-fcn([0,0]),fcn([0,1])-fcn([0,0])]));
    temp:=eigenvectors(transpose(Atemp)*Atemp);

    if (temp[2]=2 and temp[1]>=1) #uniform stretch, factor >=1
        then return(print("expands uniformly, by " , temp[1] , "in all
directions - not contraction!"));
        fi;
    if (temp[2]=2 and temp[1]<1) #uniform stretch, factor <1
        then return(print("contracts uniformly, by " , temp[1] , "in all
directions - contraction!"));
        fi;

    if (temp[1][1]>temp[2][1] and temp[1][1]>=1)
        then return(print ("not a Euclidean contraction: maximum
stretch factor" , sqrt(temp[1][1]) , "in direction" ,
temp[1][3][1] ));
        fi;
    if (temp[1][1]<temp[2][1] and temp[2][1]>=1)
        then return(print ("not a Euclidean contraction:
maximum stretch factor" , sqrt(temp[2][1]) , "in direction" ,
temp[2][3][1] ));
        fi;
    if (temp[1][1]<1 and temp[2][1]<1)
        then print ("contraction!");
        fi;
```

```
end:
```

```
>
```

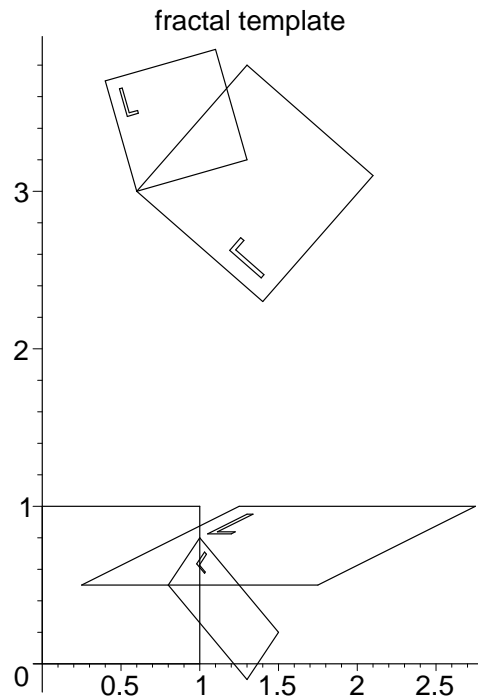
```
[ >
```

Example:

```
> f1:=P->AFFINE1(P,.7,.8,.8,-.7,.6,3);  
f2:=P->AFFINE1(P,.5,-.6,.2,.3,.8,.5);  
f3:=P->AFFINE1(P,1.5,0,1,.5,.25,.5);  
f4:=P->AFFINE1(P,.7,.2,-.2,.7,.6,3);
```

$$f1 := P \rightarrow \text{AFFINE1}(P, 0.7, 0.8, 0.8, -0.7, 0.6, 3)$$
$$f2 := P \rightarrow \text{AFFINE1}(P, 0.5, -0.6, 0.2, 0.3, 0.8, 0.5)$$
$$f3 := P \rightarrow \text{AFFINE1}(P, 1.5, 0, 1, 0.5, 0.25, 0.5)$$
$$f4 := P \rightarrow \text{AFFINE1}(P, 0.7, 0.2, -0.2, 0.7, 0.6, 3)$$

```
> TESTMAP([f1,f2,f3,f4]);
```



```
> STRETCHTEST(f1);
```

"expands uniformly, by ", 1.13, "in all directions - not contraction!"

```
> STRETCHTEST(f2);
```

"contraction!"

```
> STRETCHTEST(f3);
```

"not a Euclidean contraction: maximum stretch factor", 1.825, "in direction", [-0.8112, -0.5850]

```
[ > STRETCHTEST ( f4 ) ;  
      "contracts uniformly, by ", 0.53, "in all directions - contraction!"  
[ >  
[ >  
[ >  
[ >  
[ >
```