

The cloud of linear combinations

To look at the vector space spanned by a set of vectors it is useful to look at a cloud of linear combinations of vectors.

We want to start with some random functions that let us look at appropriate linear combinations.

```
> restart:with(plots):with(LinearAlgebra):
  rand0to1 := rand(0..1000)/1000.0:
  randneg2to2 := rand(-1000..1000)/500.0:
Warning, the name changecoords has been redefined
```

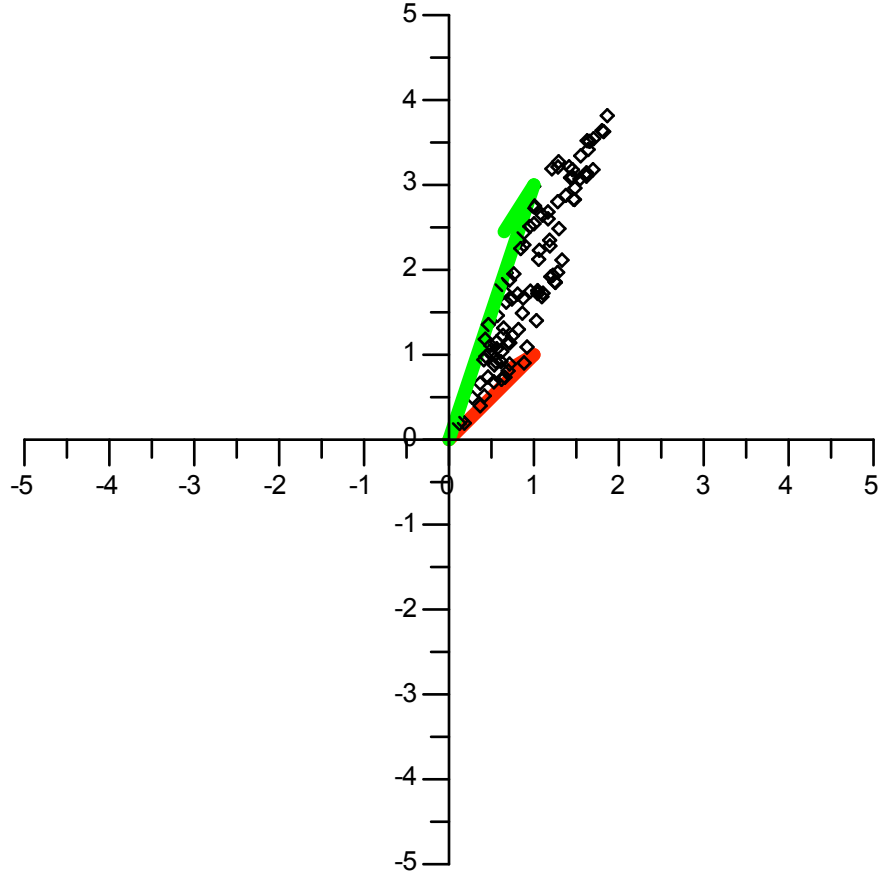
We start with some 2-vectors and 3-vectors we can use for computations.

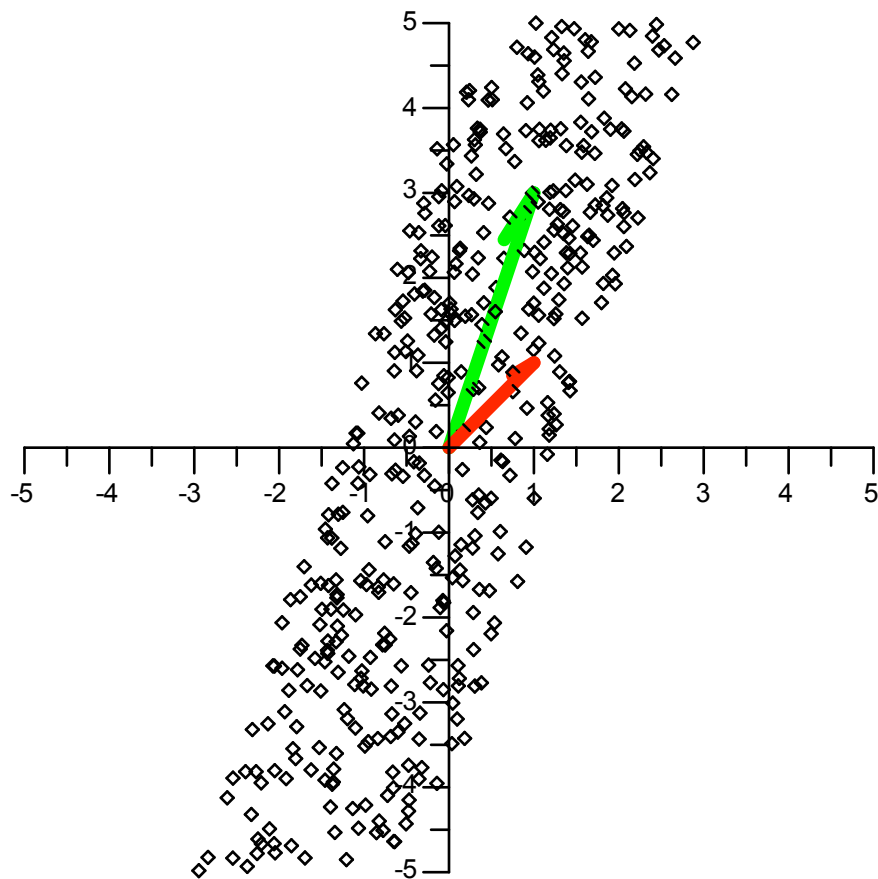
```
> v1 := <1, 1>; v2 := <1, 3>;
  w1 := <1 | 1 | 1>; w2 := <-1 | 3 | 2>; w3 := <2 | 3 | -1>;
      v1 :=  $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ 
      v2 :=  $\begin{bmatrix} 1 \\ 3 \end{bmatrix}$ 
      w1 := [ 1 1 1 ]
      w2 := [-1 3 2 ]
      w3 := [ 2 3 -1 ]
```

(1)

There are now several things to look at. In R2 we can look at a parallelogram of linear combinations bounded by the vectors. We can also look at the larger cloud of linear combinations to see a span.

```
> SpanSetPoints := {seq(rand0to1()*v1+rand0to1()*v2,i=1..100)}:
  SPoints := pointplot(SpanSetPoints,view=[-5..5,-5..5]):
  Hv1 := arrow(v1,color=red, shape=harpoon,thickness=5):
  Hv2 := arrow(v2,color=green, shape=harpoon,thickness=5):
  display([SPoints, Hv1, Hv2],scaling=constrained);
> SpanSetPoints2 := {seq(randneg2to2()*v1+randneg2to2()*v2,i=1..500)
  }:
  SPoints2 := pointplot(SpanSetPoints2,view=[-5..5,-5..5]):
  display([SPoints2, Hv1, Hv2],scaling=constrained);
```





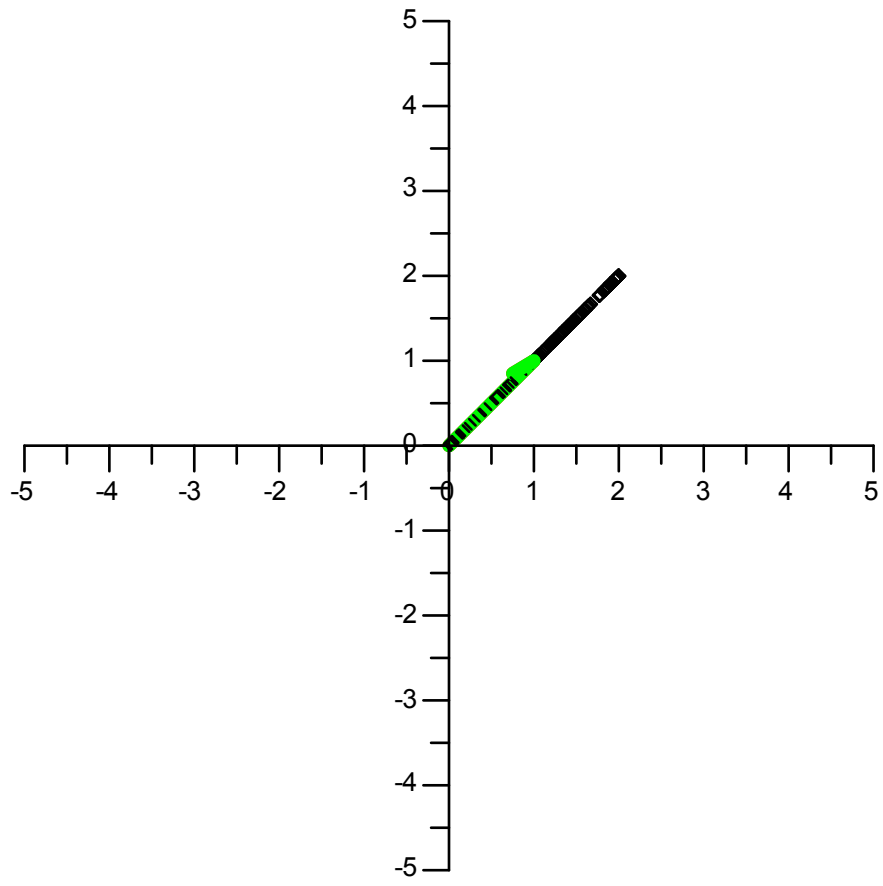
It is easy to see that the larger cloud goes everywhere in the plane.

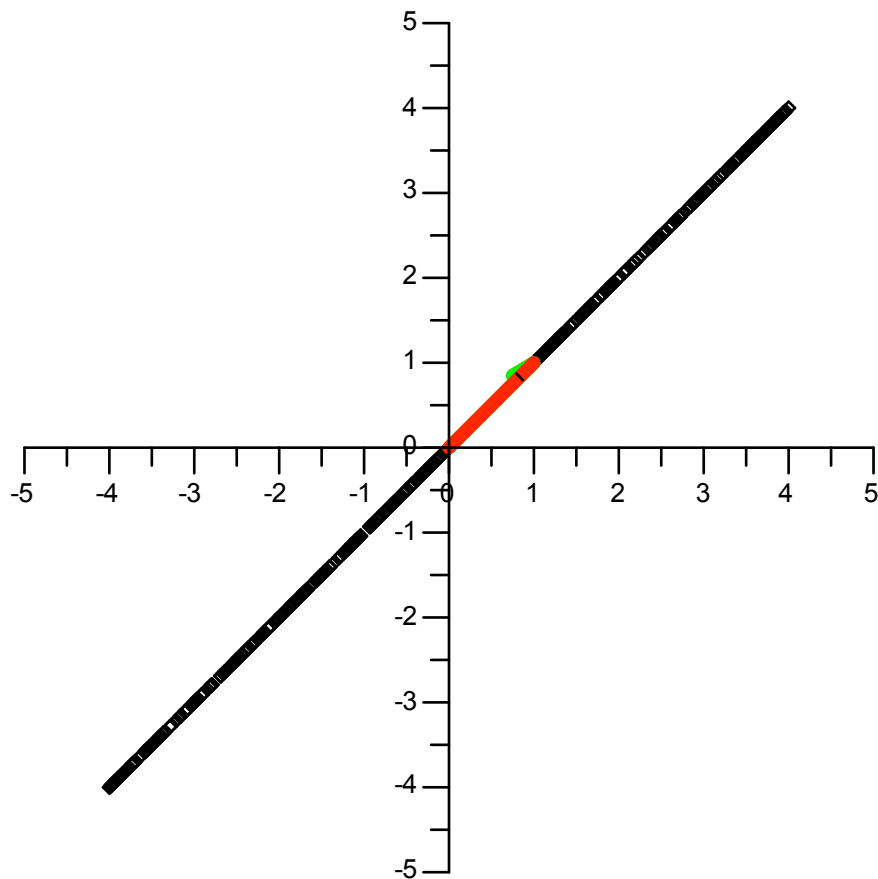
For comparison, we can try the same thing with the first vector repeated twice.

```

> SpanSetPoints := {seq(rand0to1()*v1+rand0to1()*v1,i=1..100)}:
SPoints := pointplot(SpanSetPoints,view=[-5..5,-5..5]):
Hv1 := arrow(v1,color=red, shape=harpoon,thickness=5):
Hv2 := arrow(v1,color=green, shape=harpoon,thickness=5):
display([SPoints, Hv1, Hv2],scaling=constrained);
> SpanSetPoints2 := {seq(randneg2to2()*v1+randneg2to2()*v1,i=1..500)
}:
SPoints2 := pointplot(SpanSetPoints2,view=[-5..5,-5..5]):
display([SPoints2, Hv1, Hv2],scaling=constrained);

```

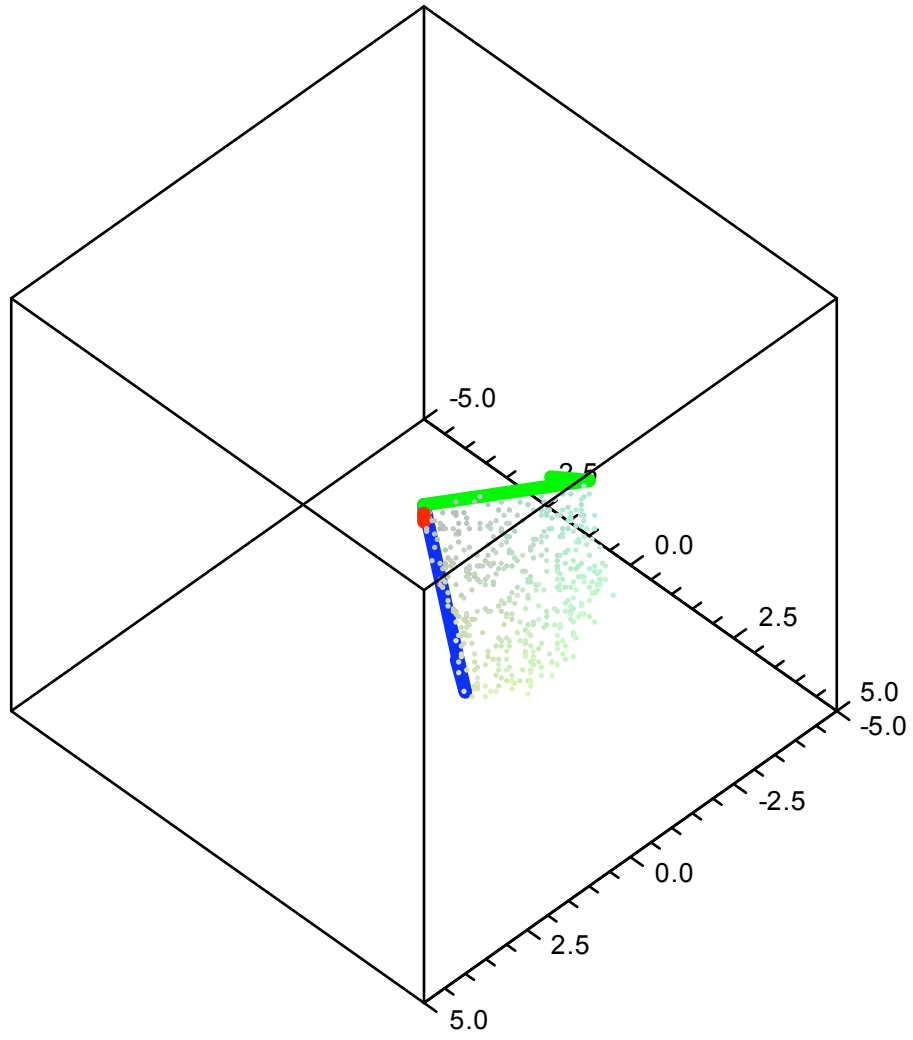


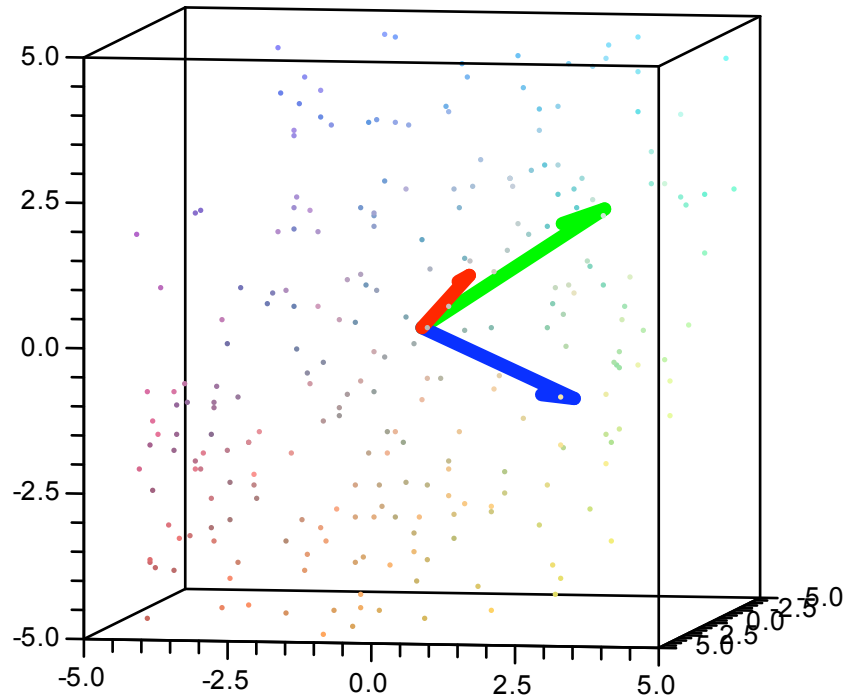


>

Now we try the same thing in 3D.

```
> SpanSetPoints := {seq(rand0to1()*w1+rand0to1()*w2+rand0to1()*w3, i=
1..500)}:
SPoints := pointplot3d(SpanSetPoints, view=[-5..5, -5..5, -5..5]):
Hw1 := arrow(w1, color=red, shape=harpoon, thickness=5):
Hw2 := arrow(w2, color=green, shape=harpoon, thickness=5):
Hw3 := arrow(w3, color=blue, shape=harpoon, thickness=5):
display3d([SPoints, Hw1, Hw2, Hw3], scaling=constrained, axes=
boxed);
SpanSetPoints2 := {seq(randneg2to2()*w1 +
randneg2to2()*w2+randneg2to2()*w3, i=1..500)}:
SPoints2 := pointplot3d(SpanSetPoints2, view=[-5..5, -5..5, -5..5]):
display3d([SPoints2, Hw1, Hw2, Hw3], scaling=constrained, axes=
boxed);
```





As before, interesting things happen when we repeat vectors and use a trivially non-independent set.

```
> SpanSetPoints := {seq(rand0to1()*w1+rand0to1()*w2+
    rand0to1()*(w1+w2),i=1..500)}:
SPoints := pointplot3d(SpanSetPoints,view=[-5..5,-5..5,-5..5]):
Hw1 := arrow(w1,color=red, shape=harpoon,thickness=5):
Hw2 := arrow(w2,color=green, shape=harpoon,thickness=5):
Hw3 := arrow(w1+w2,color=blue, shape=harpoon,thickness=5):
display3d([SPoints, Hw1, Hw2, Hw3],scaling=constrained, axes=
boxed);
SpanSetPoints2 := {seq(randneg2to2()*w1 +
    randneg2to2()*w2+randneg2to2()*(w1+w2), i=1..500)}:
SPoints2 := pointplot3d(SpanSetPoints2,view=[-5..5,-5..5,-5..5]):
display3d([SPoints2, Hw1, Hw2, Hw3],scaling=constrained, axes=
boxed);
```

