Sage and Linear Algebra Worksheet

FCLA Section MR

Robert Beezer
Department of Mathematics and Computer Science
University of Puget Sound

Spring 2017

1 A Linear Transformation and some Bases

In this section we define a linear transformation from $\mathbb{C}^6$ to $\mathbb{C}^4$. The definition is a 4×6 matrix of rank 4 that we will use to multiply input vectors with a matrix-vector product. It is not important if the linear transformation is injective and/or surjective.

```
m, n = 4, 6
A = random_matrix(QQ, m, n, algorithm='echelonizable',
    rank=m, upper_bound=9)
A
```

```
T = linear_transformation(A, side='right')
T
```

And we construct two random bases, one for the domain and one for the codomain, extracted from the columns of unimodular matrices.

```
Dmat = random_matrix(QQ, n, n, algorithm='unimodular',
    upper_bound=9)
D = Dmat.columns()
D
```

```
Cmat = random_matrix(QQ, m, m, algorithm='unimodular',
    upper_bound=9)
C = Cmat.columns()
C
```

2 A Matrix Representation, Old Style

We will coordinatize the outputs of the linear transformation, for inputs from the basis of the domain, relative to the basis of the codomain, and pack them in a matrix.

Outputs on the domain basis first.

```
outputs = [T(b) for b in D]
outputs
```

We make a vector space for the codomain, with the desired basis.

```
VC = (QQ^m).subspace_with_basis(C)
VC
```

Now, coordinate versions of the outputs.

```
coord = [VC.coordinate_vector(t) for t in outputs]
coord
```

And then we pack them into a matrix.

```
rep = column_matrix(coord)
rep
```

Does the representation "work" as it should? We need a vector space for the domain before we can check.

```
VD = (QQ^n).subspace_with_basis(D)
VD
```

OK, coordinatize input, multiply by representation matrix, un-coordinatize (linear combination). This is the fundamental Theorem FTMR at work.

```
u = random_vector(QQ, 6)
out =
    VC.linear_combination_of_basis(rep*VD.coordinate_vector(u))
u, out, T(u) ==  out
```

Nice.

3 A Matrix Representation, Sage Style

Now we do it Sage style. The matrix of the linear transformation first, as we like to see it.

```
T.matrix(side='right')
```

Now we replace the domain and codomain with new vector spaces, carrying different bases. We are not really "restricting" the domain and codomain, we are replacing them by the same vector space, but each has a different basis.

```
S = T.restrict_domain(VD).restrict_codomain(VC)
rep2 = S.matrix(side='right')
rep2
```

Bingo! This is representation we found above. A one-liner in Sage.

```
rep2 == rep
```

4 Sneak Preview

Ponder the following computation — matrix representations and nonsingular matrices with columns from the two bases.

```
S.matrix(side='right') -
    Cmat.inverse()*T.matrix(side='right')*Dmat
```

Notice how Sage is conflicted about if `S` and `T` are equal or not.

```
S == T
```

```
S.is_equal_function(T)
```