Smith Normal Forms of Matrices over Euclidean Domains

Corbin McNeill and Elias Schomer

Department of Mathematics, Wheaton College, Wheaton, IL 60187

(Dated: October 28, 2016)

I. INTRODUCTION

For all matrices $A \in M_{n\times l}(R)$, there exist $S \in GL_n(R)$ and $T \in GL_l(R)$ such that

$$SAT = \begin{pmatrix} d_1 & 0 & 0 & \cdots & & 0 \\ 0 & d_2 & 0 & \cdots & & 0 \\ 0 & 0 & \ddots & & & 0 \\ \vdots & \vdots & & d_m & & \vdots \\ & & & & \ddots & \\ 0 & 0 & & \cdots & & 0 \end{pmatrix}$$

is an $n \times l$ matrix where the nonzero entries d_i satisfy $d_i \mid d_j$ whenever $i \leq j$. Furthermore, if S' and T' are other invertible matrices such that $S'AT'$ is a diagonal matrix with entries $d'_1, \ldots, d'_s, 0, \ldots, 0$ with $d'_i \mid d'_j$ whenever $i \leq j$, then $s = m$ and d'_i is an associate of d_i. The matrix SAT is called the Smith normal form of A and S and T are the left-reducing and right-reducing matrices, respectively.

The Smith normal form of a matrix has some interesting and useful applications including finding the invariant factor decomposition of a finitely generated R-module, where R is a PID. In this project we created a program that returns the Smith normal form of a given matrix, along with the left-reducing and right-reducing matrices. The program calculates these matrices over the integers and Gaussian integers and can be readily extended to any other Euclidean domains if one creates the class and defines the operations for that Euclidean domain.

II. EXTENDED EUCLIDEAN ALGORITHM

The extended Euclidean Algorithm as extremely helpful in determining which replace row and replace column operations were necessary in the process of diagonalizing the matrix. In the typical implementation of the extended Euclidean Algorithm, one runs the Euclidean Algorithm to determine the greatest common divisor of two numbers. Backwards substitutions are made until one arrives at the linear combination of the two initial numbers that yields the greatest common divisor. We found an iterative process that allowed us to compute the linear combination in conjunction with the calculation of the greatest common divisor.

The beauty of this algorithm is that it relies solely on the use of addition, subtraction,

multiplication, floored division, and the remainder operations. So this algorithm can be used to find *gcd* and linear combinations resulting in the *gcd* for any created class with these five operations defined for them.

III. MATRIX OPERATIONS

The main procedure of the algorithm, the process which manipulates matrices into corresponding Smith normal form matrices and creates their left and right reducing matrices, functions by considering shells of the input matrix. Each ith shell contains the $[i, i]$ element of the matrix as well as all elements to the right in the same row and all elements below in the same column. The algorithm works on each shell successively by manipulating the matrix by linear combinations of rows and columns and row and column swaps in order to move a *gcd* of these elements into the $[i, i]$ position. Since this element will divide all other elements of the shell, we use it to make all other elements of the shell 0.

Once this process has been run on each shell we have a diagonal matrix; however, it is not necessarily in Smith normal form yet since the diagonal elements do not necessarily divide each other. We therefore work through each pair of adjacent elements in the diagonal and perform row and column linear combinations in order to cause the first to divide the second while maintaining the diagonal nature of the matrix.

All matrix operations are recorded in the left and right reducing matrices which are additionally returned by algorithm. This completes the Smith normal form computation.

IV. RUNNING INSTRUCTIONS

The algorithm, implemented in python 2, can be executed by the following command:

```
>> python snf.py
```

The program will guide the user through the matrix input process, perform the Smith normal form algorithm, print the results, and immediately terminate.