

**SUPPLEMENT FOR LAB2  
THE SINGULAR VALUE DECOMPOSITION  
MATH 181: COMPUTATIONAL MATHEMATICS**

R. ROBINSON

1. INTRODUCTION

Consider a system of  $n$  linear equations in  $m$  variables.

$$\begin{aligned} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,m}x_m &= b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,m}x_m &= b_2 \\ &\vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + \cdots + a_{n,m}x_m &= b_n. \end{aligned}$$

Rewrite the system as a matrix equation

$$(1) \quad \mathbf{Ax} = \mathbf{b}.$$

*Numerical Linear Algebra* is the field that solves such equations for  $\mathbf{x}$ , as well as related problems such as finding the *inverse* or *determinate* of  $A$  when  $A$  is square.

The goal is to do this as efficiently as possible ( $n$  and  $m$  may be huge) and introducing as few round off errors as possible (usually caused by dividing by a number close to zero). In particular, not being careful about division by small numbers may lead to loss of significance and instability.

MATLAB has routines to perform the Gauss-Jordan algorithm (`>> rref(A)`) and automatic routines for the inverse (`>> A ^ (-1)`) and

---

*Date:* 1/24/2001.

determinate ( $\gg \det(\mathbf{A})$ ). Here we concentrate on a more advanced method called the *singular value decomposition*. We will take a very practical approach; we will not prove the Singular Value Decomposition Theorem nor will we explain the algorithm for finding it. Instead we will show how to use MATLAB's command  $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{svd}(\mathbf{A})$ .

## 2. CLASSIFICATION OF PROBLEMS

There are essentially three cases of (1).

1.  $A$  is  $n \times n$  and  $\det(A) \neq 0$ . This is called the *nonsingular* case.

In this case we might want a single solution, solutions for several different right hand sides  $\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_k$ , or might want to find  $A^{-1}$ .

2.  $n < m$  (or  $n = m$  but  $\det(A) = 0$ , or  $n > m$  but  $\text{rank}(A) < m$ ).

We call this case *underdetermined*.

In this case we expect infinitely many solutions if there are any (which there may not be).

3. The case  $n > m$  (except as above) is called *overdetermined*. Typically in this case there is *no* solution. However, we sometimes settle for finding the vector  $\mathbf{x}$  that minimizes the error  $\|\mathbf{Ax} - \mathbf{b}\|$  (called the *residual*). Here  $\|\mathbf{y}\| = \sqrt{\mathbf{y} \cdot \mathbf{y}}$ .

This is called the *least squares* approximation of a solution.

The simplest method (taught in Linear Algebra classes) is Gauss-Jordan, based on the idea of row reduction. Beware of using this too naively though! Without introducing *pivoting* it is numerically unstable (see Press et al). A slightly more sophisticated method is the LU decomposition (see Press et al). Numerical Linear Algebra studies many other methods as well, especially for cases of *sparse* (i.e., mostly zero) matrices.

Recall that in case 1., the *homogeneous equation*  $Ax = 0$  has only a trivial solution  $\mathbf{x} = 0$  (in fact the solution is unique for any  $\mathbf{b}$ ).

In case 2., the solutions to the homogeneous equation form a subspace of  $\mathbf{R}^m$  called the *nullspace*. The dimension of the nullspace is called the *nullity* of  $A$ . Let  $k$  be the nullity of  $A$  and let  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$  be a *basis for the nullspace*. We call a linear combination

$$\mathbf{x}_h = a_1\mathbf{x}_1 + \dots + a_k\mathbf{x}_k$$

the *general solution* to the homogeneous equation.

Now we also have the *range* of  $A$ , which is

$$\{\mathbf{b} \in \mathbf{R}^n : A\mathbf{x} = \mathbf{b} \text{ for some } \mathbf{x} \in \mathbf{R}^m\}.$$

This is a subset of  $\mathbf{R}^n$ . It is just the set of all  $\mathbf{b}$  for which there is a solution to (1).

The dimension of the range of  $A$  is called the *rank* of  $A$ . One of the most important results in basic linear algebra is

**Theorem 1.** (The rank theorem)  $\text{rank}(A) + \text{nullity}(A) = m$ .

We also have the following

**Theorem 2.** Let  $x_p$  be any solution to (1) and let  $x_h$  be the general solution to the homogeneous equation. The general solution to (1) is given by  $x = x_h + x_p$ .

Now we come to the singular value decomposition.

**Theorem 3.** (Singular Value Decomposition) Let  $A$  be an  $n \times m$ . Then there exists an  $n \times m$  matrix  $U$  with  $U^T U = I$ , an  $m \times m$  matrix  $V$  with  $V^T V = I$  and a diagonal matrix  $S = \text{diag}(w_1, \dots, w_n)$  such

that

$$(2) \quad A = USV^T.$$

The diagonal elements of  $S$  are called *singular values*. The columns of  $U$  corresponding to non-zero singular values form a basis (an orthogonal basis) for the range of  $A$ . The columns of  $V$  corresponding to the zero singular values form a basis for the nullspace.

**Theorem 4.** *The vector  $x = V \cdot \text{diag}(1/w_i) \cdot (U^T \mathbf{b})$  (where we put  $1/w_i = 0$  if  $w_i = 0!$ ) is a solution to (1), if it is solvable. Otherwise it is a least squares approximate solution.*

In MATLAB you get this by typing `>> [U,S,V]=svd(A)`.

This algorithm is extremely stable and has many uses. For example:

1. We can use this to find general solutions (or unique solutions) to systems of equations.
2. We can use it to find orthonormal bases for subspaces of  $\mathbf{R}^n$ .
3. It is also useful for solving linear least squares problems.

### 3. LEAST SQUARES

Suppose we have a bunch of data points  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ , and a collection of “basic” functions  $f_1, f_2, \dots, f_m$ . We want to model the points as well as possible by a function of the form:

$$y = a_1 f_1(x) + a_2 f_2(x) + \dots + a_m f_m(x).$$

Ideally, the fit would be perfect and we would have

$$\begin{aligned} y_1 &= a_1 f_1(x_1) + a_2 f_2(x_1) + \cdots + a_m f_m(x_1) \\ y_2 &= a_1 f_1(x_2) + a_2 f_2(x_2) + \cdots + a_m f_m(x_2) \\ &\vdots \\ y_n &= a_1 f_1(x_n) + a_2 f_2(x_n) + \cdots + a_m f_m(x_n). \end{aligned}$$

That is

$$F\mathbf{a} = \mathbf{y}$$

where

$$F = \begin{pmatrix} f_1(x_1) & \cdots & f_m(x_1) \\ f_1(x_2) & \cdots & f_m(x_2) \\ \vdots & & \vdots \\ f_1(x_n) & \cdots & f_m(x_n) \end{pmatrix}$$

If there are a lot of data points the system will be overdetermined and an exact solution will be impossible. However the least squares approximation (Theorem 4) will minimize

$$(3) \quad \|\mathbf{y} - F\mathbf{a}\|^2 = \sum_{i=1}^n \left( y_i - \sum_{j=1}^m a_j f_j(x_i) \right)^2.$$

The classic case is of course  $f_1(x) = x$  and  $f_2(x) = 1$ . Letting  $a_1 = m$  and  $a_2 = b$  we find the best *line*  $y = mx + b$  through the data points!