Singular Value Decomposition

By: Ajay Rudrabhatla & Cody Peter

SVD Concept

- Break down matrix into 3 parts
  - $A = U S V^T$
  - V is derived from $A^T A$
  - U is derived from $A A^T$
  - S is derived from sqrt(eigenvalues) on diagonal
- Saves space because U, S, V are less dense than A
- A is nxm, U is nxn, S is nxm, V is mxm

Example of SVD

$$A = \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$W\,\mathbf{x} = \lambda\,\mathbf{x}$$

$$AA^T = \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 4 & 0 & 0 \\ 1 & 3 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 20 & 14 & 0 & 0 \\ 14 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = W$$

Since W $\mathbf{x} = \lambda\,\mathbf{x}$ then (W- λI) $\mathbf{x} = 0$

$$\begin{bmatrix} 20-\lambda & 14 & 0 & 0 \\ 14 & 10-\lambda & 0 & 0 \\ 0 & 0 & -\lambda & 0 \\ 0 & 0 & 0 & -\lambda \end{bmatrix} \mathbf{x} = (W - \lambda I)\mathbf{x} = 0$$

λ=0, λ=0; λ = 15+Ö221.5 ~ 29.883; λ = 15-Ö221.5 ~ 0.117

19.883 x1 + 14 x2 = 0
14 x1 + 9.883 x2 = 0
x3 = 0
x4 = 0

-9.883 x1 + 14 x2 = 0
14 x1 - 19.883 x2 = 0
x3 = 0
x4 = 0

$$U = \begin{bmatrix} 0.82 & -0.58 & 0 & 0 \\ 0.58 & 0.82 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A^T.A = \begin{bmatrix} 2 & 4 & 0 & 0 \\ 1 & 3 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$V = \begin{bmatrix} 0.40 & -0.91 \\ 0.91 & 0.40 \end{bmatrix}$$

$$S = \begin{bmatrix} 5.47 & 0 \\ 0 & 0.37 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Our Gameplan of Parallelization

- Calculate eigenvalues in parallel
- Use MPI to pass each eigenvalue to a different process
- Each process calculates an eigenvector which then returns to process 0 (openmp)
- Methods used:
  - MPI_Scatter()
  - MPI_Gather()
- Process 0 makes V^T, U, and S out of the eigenvectors and values

Code that We Found

- Code to find eigenvalues
- Code to transpose a matrix
- Code to row reduce into RREF
    - Needed for finding eigenvectors
    - Posed some difficulties

Changes We Made to the Existing Code

- Added methods to find eigenvectors
- Added MPI functions to make the code multi-process
- Added openMP

```
void subtractEigenValue(Matrix m, EL_Type eigenValue){
        int i, j;
        for(i = 0; i < m->dim_y; i++){
                EL_Type cur = m->mtx[i][i];
                m->mtx[i][i] = cur - eigenValue;
        }
}
```

Changes Cont.

```
void EigenVectorNormalize(Matrix m, EL_Type* vecBuff, EL_Type eigenValue){
	int i, j;
	EL_Type sum_of_squares = 0;
	for(i = 0; i < m->dim_y; i++){
		//printf("Augmented Value: %f\n",m->mtx[i][m->dim_x - 1]);

		EL_Type check = 1.0/m->mtx[i][m->dim_x - 1];
		if(check < 0){
			check = check * -1.0;
		}

		if(check == eigenValue || m->mtx[i][m->dim_x - 1] == 1.0){


			continue;
		}
		sum_of_squares += m->mtx[i][m->dim_x - 1] * m->mtx[i][m->dim_x - 1];
	}
	EL_Type normalization_scalar = 0.0;
	if(sum_of_squares == 0.0){
		normalization_scalar = 0.0;
	}
	else{
		normalization_scalar = 1.0/(sqrt(sum_of_squares));
	}
	printf("Norm Scalar: %f\n", normalization_scalar);
	for(j = 0; j < m->dim_y; j++){
		EL_Type check2 = 1.0/(m->mtx[j][m->dim_x - 1]);
		printf("check2: %f    %f\n", check2 * check2, eigenValue * eigenValue);

		if((((check2 * check2) - (eigenValue * eigenValue)) < .000001 && check2*check2 - eigenValue*eigenValue > -.000001) || m->mtx[j][m->dim_x-1] == 1.0 || m->mtx[j][m->dim_x-1] == -1){
		//	printf("hereeeee\n");
			vecBuff[j] = 0.0;
		}
		else{
			vecBuff[j] = normalization_scalar * m->mtx[j][m->dim_x - 1];
		}

	}

}
```

Difficulties

- Finding eigenvectors
- Making the code generic for all sizes of matrices

Serial vs. Parallel Versions

Serial Times in seconds (nxn matrix size)

100: 4.7

150: 25

200: 115

Parallel Times in seconds (nxn matrix size): 2, 4, 8, 16 nodes

100:	2.6643,	1.50775,	1.16926,	1.05103
150:	16.4239,	9.5083,	6.85412,	5.5307
200:	54.568,	28.849,	17.959,	15.11712

Conclusion

- SVD has a way that it can be efficiently parallelized

Sources

- http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm
- https://rosettacode.org/wiki/Reduced_row_echelon_form#C
- https://blog.statsbot.co/singular-value-decomposition-tutorial-52c695315254
- https://www.mpi-inf.mpg.de/fileadmin/inf/d5/teaching/ss17_dmm/lectures/2017-05-08-lin_alg_and_svd.pdf
- https://stackoverflow.com/questions/769/solving-a-linear-equation
- https://dml.cz/bitstream/handle/10338.dmlcz/702748/PANM_15-2010-1_18.pdf
- https://www.math.cuhk.edu.hk/~lmlui/CaoSVDintro.pdf
- http://www.math.utah.edu/~goller/F15_M2270/BradyMathews_SVDImage.pdf