

MATH 335: Numerical Analysis

Problem Set 7, Solutions

- AP 3.16 (a) Having stored $f(V) = (P + 3/V^2)(3V - 1) - 8T$ in an OCTAVE function, I called secant as demonstrated below

```
octave:261> secant(@f, 1/2, 1, .000005, 100)
```

iter	xn	f(xn)	f(xn+1)-f(xn)	xn+1-xn
0	0.500000	-7.000000		
0	1.000000	2.000000	9.000000	0.500000
1	0.888889	0.328125	-1.671875	0.111111
2	0.867082	-0.003140	-0.331265	0.021807
3	0.867289	0.000009	0.003148	0.000207
4	0.867288	0.000000	-0.000009	0.000001

So, the value is $V \doteq 0.867288$.

- (b) With $P = 4$ and $T = 1$, we get $V \doteq 0.500000$.

- (c) With $P = 5$ and $T = 5$, we get $V \doteq 2.811917$.

- ★8 (a) We require only one operation on the augmented matrix:

$$\left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 2 & 2 & 4 \end{array} \right] \sim \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & -2000 & -2000 \end{array} \right] \Rightarrow x_1 = 0, x_2 = 1.$$

- (b) Following the discussion on *scaled partial pivoting* in Section 4.3, we first form a vector $\mathbf{c} = (0.2, 2)$. To find our first pivot, we compare ratios $\mathbf{r} = (0.0002/0.2, 2/2) = (0.001, 1)$, and opt for an exchange of rows:

$$\left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 2 & 2 & 4 \end{array} \right] \sim \left[\begin{array}{cc|c} 2 & 2 & 4 \\ 0.0002 & 0.2 & 0.2 \end{array} \right] \sim \left[\begin{array}{cc|c} 2 & 2 & 4 \\ 0 & 0.200 & 0.200 \end{array} \right]$$

which yields $x_1 = 1, x_2 = 1$.

- (c) Before multiplication by \mathbf{S} we have

$$\left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 2 & 2 & 5.6 \end{array} \right] \sim \left[\begin{array}{cc|c} 0.0002 & 0.2 & 0.2 \\ 0 & -2000 & -1990 \end{array} \right] \Rightarrow x_1 = 5, x_2 = 0.995.$$

After multiplication by \mathbf{S} we have

$$\left[\begin{array}{cc|c} 0.002 & 2 & 2 \\ 2 & 2 & 5.6 \end{array} \right] \sim \left[\begin{array}{cc|c} 0.002 & 2 & 2 \\ 0 & -2000 & -1990 \end{array} \right] \Rightarrow x_1 = 5, x_2 = 0.995.$$

The technique does *not* help. As my graduate advisor/numerical analysis professor in graduate school said, “It’s not the scaling, it’s the pivoting strategy scaling reveals.”

- (d) The exact solution is $(-1500, -14)$. Using naive GE we have (after multiplication by \mathbf{S}):

$$\left[\begin{array}{cc|c} 0.07 & -8 & 7 \\ -0.1 & 10 & 10 \end{array} \right] \sim \left[\begin{array}{cc|c} 0.07 & -8 & 7 \\ 0 & -1.43 & 20 \end{array} \right] \Rightarrow x_1 = -1500, x_2 = -14.$$

- ★9 (a) First, I entered the commands

```
octave:88> format rat
octave:89> naiveGE([-1 2 -1 0 4 1; 1 2 0 3 0 2; 0 -3 1 1 2 4; ...
                  1 0 2 -1 3 0; 2 -2 2 -2 1 3])
ans =
```

-1	2	-1	0	4	1
0	4	-1	3	4	3
0	0	1/4	13/4	5	25/4
0	0	0	-22	-25	-38
0	0	0	0	92/11	91/11

and got the resulting output. To solve, we use back substitution:

$$\begin{aligned} \frac{92}{11}x_5 &= \frac{91}{11} &\Rightarrow x_5 &= \frac{91}{92} \doteq 0.98913. \\ -22x_4 - 25\left(\frac{91}{92}\right) &= -38 &\Rightarrow x_4 &= \frac{111}{184} \doteq 0.60326. \\ \frac{1}{4}x_3 + \frac{13}{4}\left(\frac{111}{184}\right) + 5\left(\frac{91}{92}\right) &= \frac{25}{4} &\Rightarrow x_3 &= -\frac{21}{8} = -2.625. \\ 4x_2 + \frac{21}{8} + 3\left(\frac{111}{184}\right) + 4\left(\frac{91}{92}\right) &= 3 &\Rightarrow x_2 &= -\frac{31}{23} \doteq -1.3478. \\ -x_1 - 2\left(\frac{31}{23}\right) + \frac{21}{8} + 4\left(\frac{91}{92}\right) &= 1 &\Rightarrow x_1 &= \frac{531}{184} \doteq 2.8859. \end{aligned}$$

- (b) Here is my modified routine, saved in the file `naiveGEmod.m`:

```
function soln = naiveGEmod(A, b)
numRows = size(A)(1); numCols = size(A)(2);
if (numRows == numCols)
    ndims = numRows;
```

```

aug = [A b];
for currIdx = 1:(ndims - 1)
    pivot = aug(currIdx, currIdx);
    for ii = (currIdx+1):ndims
        aug(ii, currIdx+1:ndims+1) = aug(ii, currIdx+1:ndims+1) - ...
            aug(ii,currIdx) / pivot * aug(currIdx, currIdx+1:ndims+1);
        aug(ii, currIdx) = 0;
    end
end

A = aug(1:ndims, 1:ndims);
b = aug(1:ndims, ndims + 1);
soln = [zeros(ndims-1, 1); b(ndims)/A(ndims,ndims)];
for jj = (ndims - 1):(-1):1
    soln(jj) = (b(jj)-sum( A(jj,(jj+1):ndims)' .* soln(jj+1:ndims) ))/A(jj,jj);
end

else
    disp('Your matrix must be square.')
end
end
end

```

(c) Here is my modified routine, saved in the file `geSolve.m`:

```

function soln = geSolve(A, b)
    numRows = size(A)(1);
    numCols = size(A)(2);
    if (numRows == numCols)
        ndims = numRows;

        scales = max(abs(A'))';
        aug = [A b];
        rowSeq = 1:ndims;
        numColsProcessed = 0;
        while (numColsProcessed < ndims - 1)
            jj = numColsProcessed + 1;
            ratioVec = abs(aug(rowSeq(jj:ndims), jj)) ./ scales(rowSeq(jj:ndims));
            offset = find(ratioVec == max(ratioVec))(1) - 1;
            if (offset > 0)
                saveVal = rowSeq(jj);
                rowSeq(jj) = rowSeq(jj + offset);
            end
            numColsProcessed = numColsProcessed + 1;
        end
        soln = zeros(ndims, 1);
        soln(rowSeq(jj)) = b(rowSeq(jj));
    else
        error('Matrix must be square');
    end
end

```

```

        rowSeq(jj + offset) = saveVal;
    end

    pivot = aug(rowSeq(jj), jj);
    for ii = rowSeq(jj+1:ndims)
        aug(ii, jj+1:ndims+1) = aug(ii, jj+1:ndims+1) - ...
            aug(ii,jj) / pivot * aug(rowSeq(jj), jj+1:ndims+1);
        aug(ii, jj) = 0;
    end
    numColsProcessed = numColsProcessed + 1;
end

A = aug(1:ndims, 1:ndims);
b = aug(1:ndims, ndims + 1);
lastxEqn = rowSeq(ndims)
soln = [zeros(ndims-1, 1); b(lastxEqn)/A(lastxEqn, ndims)]
for jj = (ndims - 1):(-1):1
    idx = rowSeq(jj);
    soln(jj) = (b(idx) - sum(A(idx, (jj+1):ndims)' .* soln(jj+1:ndims))) ...
        / A(idx,jj);
end

else
    disp('Your matrix must be square.')
end
end
end

```

★10 First, we solve $\mathbf{L}\mathbf{y} = (2, 0, 2)$ to get $\mathbf{y} = (y_1, y_2, y_3) = (2, -2, 0)$. Then we solve $\mathbf{U}\mathbf{x} = \mathbf{y}$ to get $\mathbf{x} = (5, -2, 0)$.