CSS 455 Winter 2012 C. Jackels

Activity No. 4 Answers January 23, 2012

Names (must be present):

Part 1.

$$y_i = \sum_{k=1}^{n} a_{ik} x_k$$

- Given the matrix-vector product
  y = **Ax**
  where **y** is an $(r \times 1)$ column vector, **x** is an $(n \times 1)$ and **A** is an $(r \times n)$ matrix, write pseudo code for a suitable algorithm.
- Estimate in terms of *r* and *n* the number of floating point operations (multiplies, additions and subtractions) in this algorithm.

```
For i from 1 to r
  Sum = 0
  For k from 1 to n
     Sum = sum + a(i,k)*x(k)
  End
     y(i) = sum
end
```

The inner loop is run *rxn* times (an n^2 operation) and executes one add and one multiply each time: 2r n flops.

Part II

Given the algorithm for the row-ordered matrix vector product:

```
%compute the Ax product
y = zeros(m,1);
for i = 1:m
   y(i) = 0;
   for j = 1:n
      y(i) = y(i) + A(i,j)*x(j);
   end
end
```

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ 0 & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ 0 & 0 & a_{33} & a_{34} & a_{35} & a_{36} \\ 0 & 0 & 0 & a_{44} & a_{45} & a_{46} \\ 0 & 0 & 0 & 0 & a_{55} & a_{56} \\ 0 & 0 & 0 & 0 & 0 & a_{66} \end{pmatrix}$$

Modify as needed for the case where **A** is *upper triangular*

```
%compute the Ax product
y = zeros(m,1);
for i = 1:m
   y(i) = 0;
   for j = i:n
      y(i) = y(i) + A(i,j)*x(j);
   end
end
```

Each run of the j-loop (across a row) now need only begin with the diagonal element to skip multiplies involving zeros. You could write the inner loop as a dot product with the active vectors being (i:n) in length.

See reverse side

Part III

$$C_{kj} = \sum_{i=1}^{r} A_{ki} B_{ij}$$

- Given the matrix-matrix product
  C=AB
  where **A** is an ($m \times r$) matrix, **B** is an ($r \times n$) matrix and **C** is an ($m \times n$) matrix write pseudo code for a suitable algorithm.
- Estimate in terms of m, n, and r the number of floating point operations (multiplies, additions and subtractions) in this algorithm.

```
For j from 1 to n
  For k from 1 to m
      Sum = 0
      For i from 1 to r
          Sum = sum + A(k,i)*B(i,j)
      End
      C(k,j) = sum
   End
end
```

The inner loop is run rxnxn times (an n^3 operation) and executes one add and one multiply each time: 2rnm flops.

See reverse side