CS 4424
The exponent of linear algebra
Main idea

Roughly. All problems of linear algebra are more or less equivalent.

More precisely

- the exponent of a problem $P$ (multiplication, inversion, ...) is a number $\omega_P$ such that one can solve problem $P$ for matrices of size $n$ in time $O(n^{\omega_P})$.

- then

\[
\omega_{\text{product}} = \omega_{\text{inversion}} = \omega_{\text{determinant}} = \cdots
\]
Inversion $\implies$ multiplication

Suppose we want to multiply two matrices $A$ and $B$, but all that we have is an algorithm for inverse

Define

$$D = \begin{bmatrix}
I_n & A & 0 \\
0 & I_n & B \\
0 & 0 & I_n
\end{bmatrix}$$

Then

$$D^{-1} = \begin{bmatrix}
I_n & -A & AB \\
0 & I_n & -B \\
0 & 0 & I_n
\end{bmatrix}$$

So product in size $n$ can be done using inverse in size $3n$. 
Suppose we want to invert a matrix $A$, of size $n = 2^k$.

We cut $A$ into blocks

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix}.$$ 

Define $S = A_{2,2} - A_{2,1}A_{1,1}^{-1}A_{1,2}$, and suppose that $S$ is invertible. Then

$$A = \begin{bmatrix} I_m & 0 \\ A_{2,1}A_{1,1}^{-1} & I_m \end{bmatrix} \begin{bmatrix} A_{1,1} & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} I_m & A_{1,1}^{-1} \\ 0 & I_m \end{bmatrix},$$

with $m = n/2$. 


Multiplication $\Rightarrow$ inversion

Invert $A$:

\[
A^{-1} = \begin{bmatrix}
I_m & -A_{1,1}^{-1} \\
0 & I_m
\end{bmatrix}
\begin{bmatrix}
A_{1,1}^{-1} & 0 \\
0 & S^{-1}
\end{bmatrix}
\begin{bmatrix}
I_m & 0 \\
-A_{2,1}A_{1,1}^{-1} & I_m
\end{bmatrix}.
\]

Complexity:

\[
I(n) \leq 2I(n/2) + Cn^\omega
\]

implies

\[
I(n) \leq C'n^\omega
\]

Remark: this also gives the determinant.
Multiplication $\implies$ minimal polynomial

**Reminder:** the minimal polynomial of $A$ is the monic polynomial

$$p = p_0 + \cdots + p_{d-1}X^{d-1} + X^d$$

of smallest degree such that

$$p_0I_n + \cdots + p_{n-1}A^{d-1} + A^d = 0.$$

**Sparse matrix algorithm:**

- compute $u$, $Au$, $A^2u$, …
- compute $vu$, $vAu$, $vA^2u$, …
- use rational reconstruction

Does not exploit matrix multiplication: $O(n^3)$ for a dense matrix
Multiplication $\implies$ minimal polynomial

Using matrix multiplication:

- compute $Au$ and $A^2$
- compute $A^2[u \ Au] = [A^2u \ A^3u]$ and $A^4 = (A^2)^2$
- compute $A^4[u \ Au \ A^2u \ A^3u]$ and $A^8 = (A^4)^2$
- $\ldots$

This gives $O(\log(n))$ matrix products for all $u, Au, A^2u, \ldots A^n u$

The rest of the algorithm is $O(n^2)$
Automatic differentiation
Partial derivatives

If $F(X_1, \ldots, X_n)$ is a polynomial in $n$ variables, we define the partial derivatives

$$\frac{\partial F}{\partial X_1}, \ldots, \frac{\partial F}{\partial X_n},$$

where

$$\frac{\partial F}{\partial X_i}$$

is obtained by keeping all other $X_j$ constant, and differentiating in $X_i$.

Example: with

$$F = X_1X_2 - X_3X_4,$$

we get

$$\frac{\partial F}{\partial X_1} = X_2, \quad \frac{\partial F}{\partial X_2} = X_1, \quad \frac{\partial F}{\partial X_3} = -X_4, \quad \frac{\partial F}{\partial X_4} = -X_3.$$
Automatic differentiation

Prop.

• If $F$ can be computed using $L$ operations $+,-,\times$, then all partial derivatives
  \[
  \frac{\partial F}{\partial X_1}, \ldots, \frac{\partial F}{\partial X_n},
  \]
  can be computed using $5L$ operations.

• Independent of $n$!

Remarks

• widely used for optimization (using Newton’s iteration in several variables)

• some polynomials (such as $(X - 1)^n$) can be computed by a short program, even though they have a lot of monomials; usually, it is not the case.
A naive solution

We are given a program $\Gamma$ with input variables $X_1, \ldots, X_n$.

Example:

\[
\begin{align*}
G_1 &= X_1 - X_2 & G_1 &= X_1 - X_2 \\
G_2 &= G_1^2 & G_2 &= (X_1 - X_2)^2 \\
G_3 &= G_2 X_3 & G_3 &= (X_1 - X_2)^2 X_3
\end{align*}
\]

computes $(X_1 - X_2)^2 X_3$, with $L = 3$.

We can follow line-by-line and apply the rules for differentiation. This is called the direct mode.

\[
\begin{array}{c|c|c|c}
G_i & \frac{\partial G_i}{\partial X_1} & \frac{\partial G_i}{\partial X_2} & \frac{\partial G_i}{\partial X_3} \\
\hline
G_1 = X_1 - X_2 & 1 & -1 & 0 \\
G_2 = G_1^2 & 2G_1 \frac{\partial G_1}{\partial X_1} & 2G_1 \frac{\partial G_1}{\partial X_2} & 2G_1 \frac{\partial G_1}{\partial X_3} \\
G_3 = X_3 G_2 & X_3 \frac{\partial G_2}{\partial X_1} & X_3 \frac{\partial G_2}{\partial X_2} & X_3 \frac{\partial G_2}{\partial X_3} + G_2 \\
\end{array}
\]

Total: $O(nL)$
The reverse mode

Setup.

- Let $G_1, \ldots, G_L$ be the polynomials computed by $\Gamma$.
- Let $\Delta$ the program in variables $X_1, \ldots, X_n, Y$ obtained by removing the first line of $\Gamma$ and replacing $G_1$ by $Y$. Let $D_2, \ldots, D_L$ be the polynomials it computes.

Example With $\Gamma$ given by

\[
\begin{align*}
G_1 &= X_1 \times X_2 \\
G_2 &= G_1 + X_1 \\
G_3 &= G_1 \times G_2 \\
\end{align*}
\]

We get $\Delta$ given by

\[
\begin{align*}
D_2 &= Y + X_1 \\
D_3 &= Y \times D_2 \\
\end{align*}
\]
The reverse mode

Prop.  \( G_L = D_L(X_1, \ldots, X_n, G_1(X_1, \ldots, X_n)) \)
The reverse mode

Prop. \( G_L = D_L(X_1, \ldots, X_n, G_1(X_1, \ldots, X_n)) \)

Corollary For all \( i \),

\[
\frac{\partial G_L}{\partial X_i} = \frac{\partial D_L}{\partial X_i}(X_1, \ldots, X_n, G_1) + \frac{\partial D_L}{\partial Y}(X_1, \ldots, X_n, G_1) \frac{\partial G_1}{\partial X_i}.
\]
The reverse mode

Prop. \( G_L = D_L(X_1, \ldots, X_n, G_1(X_1, \ldots, X_n)) \)

Corollary For all \( i \),
\[
\frac{\partial G_L}{\partial X_i} = \frac{\partial D_L}{\partial X_i}(X_1, \ldots, X_n, G_1) + \frac{\partial D_L}{\partial Y}(X_1, \ldots, X_n, G_1) \frac{\partial G_1}{\partial X_i}.
\]

Key remark. \( G_1 \) has one of the following shapes
\[
X_a + X_b, \quad X_aX_b, \quad \lambda X_a, \quad \lambda + X_a, \quad \lambda.
\]

Hence, for \( i \notin \{a, b\} \), \( \frac{\partial G_L}{\partial X_i} = \frac{\partial D_L}{\partial X_i} \) and for \( i \in \{a, b\} \), \( \frac{\partial G_L}{\partial X_i} \) can be deduced from \( \frac{\partial D_L}{\partial X_i} \) and \( \frac{\partial D_L}{\partial Y} \) in \( O(1) \) operations.
The reverse mode

Prop. \( G_L = D_L(X_1, \ldots, X_n, G_1(X_1, \ldots, X_n)) \)

Corollary For all \( i \),

\[
\frac{\partial G_L}{\partial X_i} = \frac{\partial D_L}{\partial X_i}(X_1, \ldots, X_n, G_1) + \frac{\partial D_L}{\partial Y}(X_1, \ldots, X_n, G_1) \frac{\partial G_1}{\partial X_i}.
\]

Key remark. \( G_1 \) has one of the following shapes

\[ X_a + X_b, \ X_a X_b, \ \lambda X_a, \ \lambda + X_a, \ \lambda. \]

Hence, for \( i \notin \{a, b\} \), \( \frac{\partial G_L}{\partial X_i} = \frac{\partial D_L}{\partial X_i} \) and for \( i \in \{a, b\} \), \( \frac{\partial G_L}{\partial X_i} \) can be deduced from \( \frac{\partial D_L}{\partial X_i} \) and \( \frac{\partial D_L}{\partial Y} \) in \( O(1) \) operations.

Conclusion Suppose we know a program \( \Delta' \) that augments \( \Delta \) by computing all partial derivatives of \( D_3 \).

Then we can deduce a program \( \Gamma' \) of length \( L(\Delta') + O(1) \), that computing all partial derivatives of \( G_3 \).
Example

We detail the previous example. Removing the first instruction in $\Delta$ gives the program

$$\Phi \ E_3 = Y \times Z \quad | \quad E_3(X_1, X_2, Y, Z) = YZ.$$ 

Hence,

$$\frac{\partial E_3}{\partial X_1, X_2} = 0, \quad \frac{\partial E_3}{\partial Y} = Z, \quad \frac{\partial E_3}{\partial Z} = Y$$

So the program $\Phi'$ computes $E_3$ and its gradient:

$$\Phi' \mid \begin{align*} 
e_3 &= Y \times Z \\
e_3, X_{1,2} &= 0 \\
e_3, Y &= Z \\
e_3, Z &= Y \end{align*}$$
Example

Recall that $D_3(X_1, X_2, Y) = E_3(X_1, X_2, Y, Y + X_1)$, so

$$\frac{\partial D_3}{\partial X_1, X_2, Y} = \frac{\partial E_3}{\partial X_1, X_2, Y}(X_1, X_2, Y, Y + X_1) + \frac{\partial E_3}{\partial Z}(X_1, X_2, Y, Y + X_1) \frac{\partial (Y + X_1)}{\partial X_1, X_2, Y}$$
Example

Recall that \( D_3(X_1, X_2, Y) = E_3(X_1, X_2, Y, Y + X_1) \), so

\[
\frac{\partial D_3}{\partial X_1, X_2, Y} = \frac{\partial E_3}{\partial X_1, X_2, Y}(X_1, X_2, Y, Y + X_1) + \frac{\partial E_3}{\partial Z}(X_1, X_2, Y, Y + X_1) \frac{\partial (Y + X_1)}{\partial X_1, X_2, Y}
\]

and thus

\[
\frac{\partial D_3}{\partial X_1} = \frac{\partial E_3}{\partial X_1}(X_1, X_2, Y, Y + X_1) + \frac{\partial E_3}{\partial Z}(X_1, X_2, Y, Y + X_1)
\]

\[
\frac{\partial D_3}{\partial X_2} = \frac{\partial E_3}{\partial X_2}(X_1, X_2, Y, Y + X_1)
\]

\[
\frac{\partial D_3}{\partial Y} = \frac{\partial E_3}{\partial Y}(X_1, X_2, Y, Y + X_1) + \frac{\partial E_3}{\partial Z}(X_1, X_2, Y, Y + X_1)
\]
Example

Recall that \( D_3(X_1, X_2, Y) = E_3(X_1, X_2, Y, Y + X_1) \), so

\[
\frac{\partial D_3}{\partial X_1, X_2, Y} = \frac{\partial E_3}{\partial X_1, X_2, Y}(X_1, X_2, Y, Y + X_1) + \frac{\partial E_3}{\partial Z}(X_1, X_2, Y, Y + X_1) \frac{\partial (Y + X_1)}{\partial X_1, X_2, Y}
\]

and thus

\[
\frac{\partial D_3}{\partial X_1} = \frac{\partial E_3}{\partial X_1}(X_1, X_2, Y, Y + X_1) + \frac{\partial E_3}{\partial Z}(X_1, X_2, Y, Y + X_1)
\]

\[
\frac{\partial D_3}{\partial X_2} = \frac{\partial E_3}{\partial X_2}(X_1, X_2, Y, Y + X_1)
\]

\[
\frac{\partial D_3}{\partial Y} = \frac{\partial E_3}{\partial Y}(X_1, X_2, Y, Y + X_1) + \frac{\partial E_3}{\partial Z}(X_1, X_2, Y, Y + X_1)
\]

yielding the program \( \Delta' \)

\[
\begin{align*}
    d_2 &= Y + X_1 \\
    d_3 &= Y \times d_2 \\
    e_{3, X_{1, 2}} &= 0 \\
    e_{3, Y} &= d_2 \\
    e_{3, Z} &= Y \\
    d_{3, X_1} &= e_{3, X_{1, 2}} + e_{3, Z} \\
    d_{3, Y} &= e_{3, Y} + e_{3, Z}
\end{align*}
\]
Example

Recall that $G_3(X_1, X_2) = E_3(X_1, X_2, X_1 X_2)$, so

$$
\frac{\partial G_3}{\partial X_1} = \frac{\partial D_3}{\partial X_1}(X_1, X_2, X_1 X_2) + \frac{\partial D_3}{\partial Y}(X_1, X_2, X_1 X_2) \frac{\partial X_1 X_2}{\partial X_1}
$$

$$
= \frac{\partial D_3}{\partial X_1}(X_1, X_2, X_1 X_2) + X_2 \frac{\partial D_3}{\partial Y}(X_1, X_2, X_1 X_2)
$$

$$
\frac{\partial G_3}{\partial X_2} = \frac{\partial D_3}{\partial X_2}(X_1, X_2, X_1 X_2) + \frac{\partial D_3}{\partial Y}(X_1, X_2, X_1 X_2) \frac{\partial X_1 X_2}{\partial X_2}
$$

$$
= \frac{\partial D_3}{\partial X_2}(X_1, X_2, X_1 X_2) + X_1 \frac{\partial D_3}{\partial Y}(X_1, X_2, X_1 X_2)
$$
Example

This finally yields

\[ \Gamma' \]

\[
\begin{align*}
  g_1 &= X_1 \times X_2 \\
  g_2 &= g_1 + X_1 \\
  g_3 &= g_1 \times g_2 \\
  e_{3,x_{1,2}} &= 0 \\
  e_{3,y} &= g_2 \\
  e_{3,z} &= g_1 \\
  d_{3,x_1} &= e_{3,x_{1,2}} + e_{3,z} \\
  d_{3,y} &= e_{3,y} + e_{3,z} \\
  \text{tmp}_1 &= d_{3,y} \times X_2 \\
  g_{3,x_1} &= d_{3,x_1} + \text{tmp}_1 \\
  \text{tmp}_2 &= d_{3,y} \times X_1 \\
  g_{3,x_2} &= e_{3,x_{1,2}} + \text{tmp}_2
\end{align*}
\]
Back to matrix computations

Using automatic differentiation, an algorithm for the determinant gives an algorithm for inverse.

Prop. Let $A = [a_{i,j}]$ be a matrix of size $n$, whose entries are variables.

- The derivatives of the determinant of $A$ w.r.t. $a_{1,1}, \ldots, a_{n,n}$ are (almost) the entries of $A^{-1}$.

“Proof”. On an example: $n = 3$. Take

$$A = \begin{bmatrix}
a_{1,1} & a_{1,2} & a_{1,3} \\
a_{2,1} & a_{2,2} & a_{2,3} \\
a_{3,1} & a_{3,2} & a_{3,3}
\end{bmatrix}$$

so

$$\det(A) = a_{1,1}a_{2,2}a_{3,3} - a_{1,1}a_{2,3}a_{3,2} + a_{2,1}a_{3,2}a_{1,3} - a_{2,1}a_{1,2}a_{3,3} + a_{3,1}a_{1,2}a_{2,3} - a_{3,1}a_{2,2}a_{1,3}.$$
Take the partial derivatives:

\[
\frac{\partial A}{\partial a_{1,1}} = a_{2,2}a_{3,3} - a_{2,3}a_{3,2} \\
\frac{\partial A}{\partial a_{1,2}} = a_{3,1}a_{2,3} - a_{1,2}a_{3,3} \\
\frac{\partial A}{\partial a_{1,3}} = a_{2,1}a_{3,2} - a_{3,1}a_{2,2},
\]

whereas the entries of \( B = A^{-1} \) are

\[
b_{1,1} = \frac{a_{2,2}a_{3,3} - a_{2,3}a_{3,2}}{\det(A)} \\
b_{2,1} = \frac{a_{3,1}a_{2,3} - a_{1,2}a_{3,3}}{\det(A)} \\
b_{3,1} = \frac{a_{2,1}a_{3,2} - a_{3,1}a_{2,2}}{\det(A)}.
\]