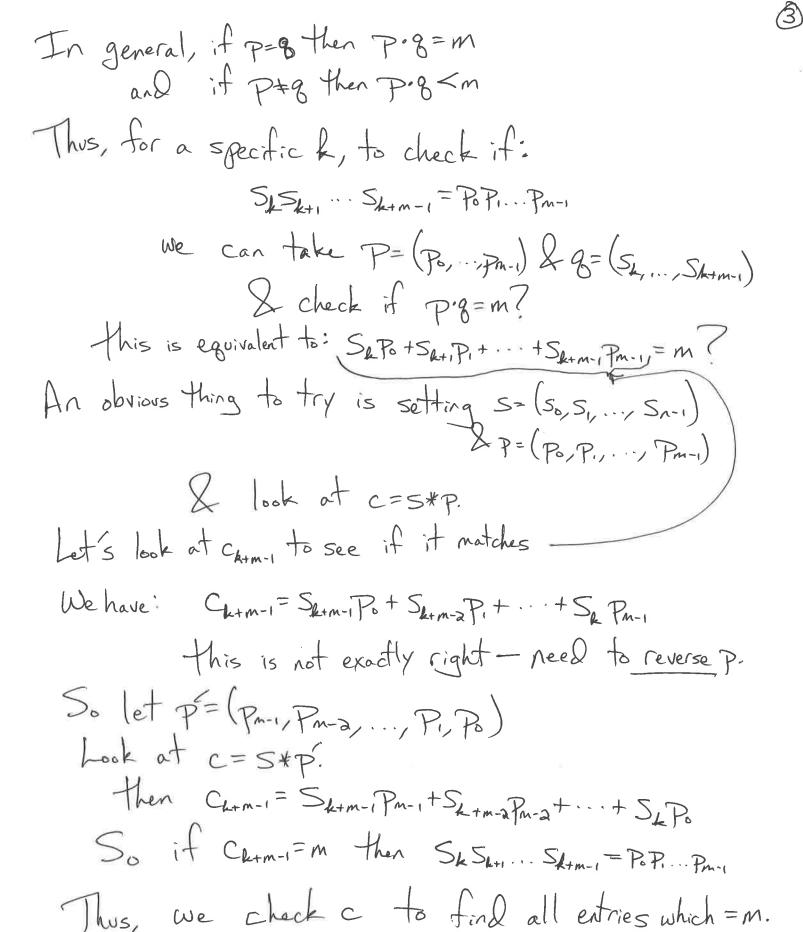
Polynomial multiplication:
Consider 2 Polynomials A(x) & B(x) where:
$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_0 x^2$
&B(x)=b.+b,x+bax2++bax2
We want to compute their product polynomial:
$C(x) = A(x)B(x) = C_0 + C_1x + C_2x^2 + \cdots + C_2x^2$
where for 1 <k=20< td=""></k=20<>
Ch= aobe+abb-1+···+abbo
Here's an example:
Here's an example. $A(x) = 1 + 2x + 3x^2 R B(x) = 2 - x + 4x^2$
then C(x) = A(x)B(x) = 2+3x + 8x2+5x3+12x4
We'll take as input the two vectors of coefficients:
$\alpha = (a_0, a_1, \dots, a_d) \times b = (b_0, b_1, \dots, b_d)$
and our goal is to compute the vector
$C = (C_0, C_0, C_0)$
The vector c is called the convolution of a & b which is denoted as: c=a*b.
Denoted as: c=a*b.
Naive approach: O(k) time for Ck > O(Q2) total time
Better approach: Using FFT = fast Forrier transform,
Better approach: Using FFT = fast Forrier transform, we'll see it can be lone in O(dlogd) time. It's a beautiful divide & conquer algorithm.

```
Some applications:
 - Pattern matching:
     Given binary strings s= S.S... Sn.1 (string)
                   & P=PoPi...Pm-1 (Pattern)
                   where n ≥ m
Goal: Find all occurrences of Pin S.
  In other words find all positions & where:
                Sh Sh+1 ... Sh+m-1 = PoP... Pm-1
 Example: 5= aaabbabbabbaa P=abbab
 Idea: mapa>-1 & b>>+1
    Then in our example: S=(-1,-1,-1,1,-1,1,-1,1,-1,1)
                 & P=(-1,1,1,-1,1)
   Consider 2 strings Plg of length My
          Say P=(-1,1,1,-1,1) & g=(-1,-1,1)
            then P.8=1-1+1+1=3
```

We get +1 if they agree in that coordinate 2-1 if they differ



(F)

Linear filtering: Replace a laterpoint by a linear combination of neighboring points.

Used for: reducing noise, adding effects etc.

Examples:

Mean filter: (also called box smoothing or moving average)

replace a point by the average of itself &

the 2m neighboring points (where Mis)

so for Qata Y=(Y1,..., Yn)

we replace it by: Yi= \frac{1}{2m+1} \sum_{i=-m} \times it

This can be computed by taking the convolution of

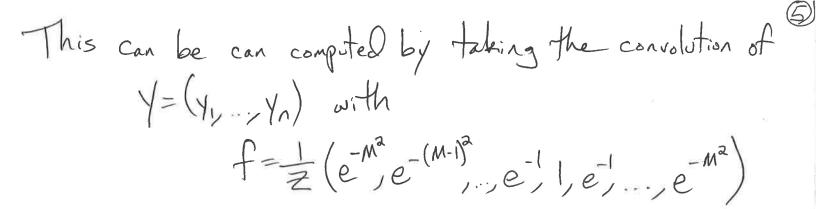
Y=(Y1,..., Yn) with f=\frac{1}{2m+1} (1,1,...,1) = (\frac{1}{2m+1} \ldots \cdots \frac{1}{2m+1} \ldots \cdots \cdots \frac{1}{2m+1} \ldots \cdots \frac{1}{2m+1} \ldots \cdots \frac{1}{2m+1} \ldots \cdots \cdots

Gaussian filter: reduce out liers by replacing datapoints
by "Gaussian average" of neighboring Points.

So for Y=(Y1,..., Yn)

replace it by: $\hat{Y}_{i} = \frac{1}{Z} \sum_{i=-M}^{M} e^{i\hat{z}} \hat{Y}_{i+i}$

where $Z = \sum_{i=-m}^{m} e^{-iz^2}$ is just a normalizing factor.

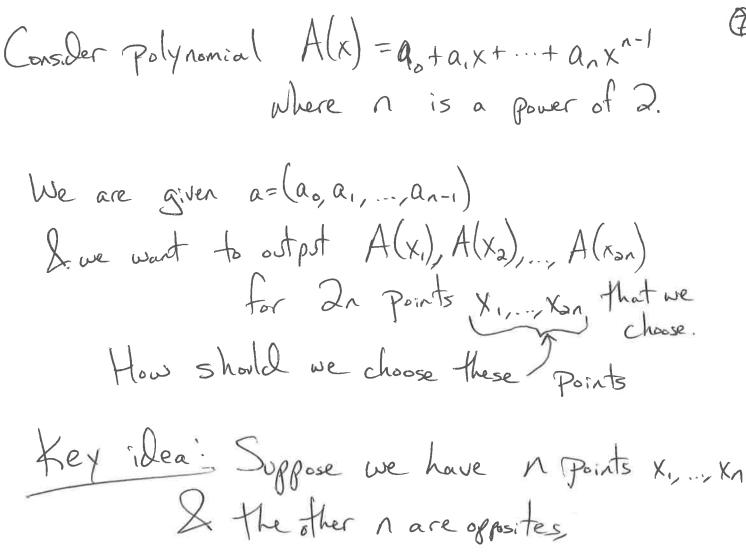


Gaussian Blur: 2-Dimensional Gaussian smoothing applied to an image to blur it.

Mary other examples, widely used.

Back to multiplying Polynomials: Two ways to represent a polynomial A(x)=ao+a,x+...+agxd 1) Coefficients: a, a, ..., a) or 2) Values: A(xo), A(xi),..., A(xd) Lenna: A degree of Polynomial is uniquely characterized by its values at any doldistinct points example: a line has D=1 & is Defined by any two points on the line. We assume the input/output is in coefficients representation, but the values representation is useful for multiplying Polynomials. Given A(x0), A(x1), A(x20) &B(x0),B(x1),...B(x20) Then C(xi)=A(xi)B(xi) for i=0,1,...,20 & this defines C(x). Need to convert between: coefficients => Values

FFT: does this conversion for carefully chosen set of Points.



So: Xn+1=-X1 Xn+2=-X2, ... Xan=-Xn Look at $A(X_i)$ & $A(-X_i) = A(X_{n+i})$ -same for even terms $a_{2k}x^{2k}$ - opposite for old terms azer x2l+1 Thus split A(x) into even & odd terms. Let aeven = (ao, aa, a4, ..., an-a) Aeven (y) = ao+aay+ayy2+...+an-ay $\begin{cases} a_{0}Q = (a_{1}, a_{3}, a_{5}, ..., a_{n-1}) \\ A_{0}Q (y) = a_{1} + a_{3}y + a_{5}y^{2} + ... + a_{n-1}y^{2} \end{cases}$

Observe that $A(x) = Aeven(x^2) + x Aodo(x^2)$ Example: A(x) = 5-3x+4x3-2x4+7x5+6x6-x7 aeven=(5,0,-2,6), aodd=(-3,4,7,-1)Aeven(y)=5-2y2+6y3, A.QQ(y)=-3+4y+7y2-y3 $A(x_i) = A_{even}(x_i^2) + X_i A_{eQQ}(x_i^2)$ $A(x_{n+1}) = A(-x_i) = Aeven(x_i^2) - x_i Aodo(x_i^2)$ So given Aeven(Yi), Aeven(Yn) L A.20 (41), ..., A.00 (4n) where $Y_1 = X_1^2, \dots, Y_n = X_n^2$ then in O(n) time we get: A(x1), ..., A(xn), A(xn), ..., A(xn) $A(-x_1)$ $A(-x_n)$ Note Aeven(y) & Aodd(y) are of degree $\frac{n-2}{2} = \frac{1}{2} - 1$ whereas A(x) has degree n-1. So to get A(x) of deg. n-1 at 2 apoints we need Aeven (y) & Assoly) each of deg. 5-1 Divide & conquer! $T(n) = 2T(\frac{\Delta}{2}) + O(n) = O(n\log n).$

But what about the next level of recursion? we have $\chi^2, \chi^2, ..., \chi^2$ Level of recursion? Level of recursion?

 $\chi^{2} = -\chi^{2}_{\frac{1}{2}+1}$ $\chi^{2}_{3} = -\chi^{2}_{3}+2$ $\chi^{3}_{5} = -\chi^{2}_{5}$

But this is impossible because x2=0 2 x2+1=0

Unless we use complex numbers.