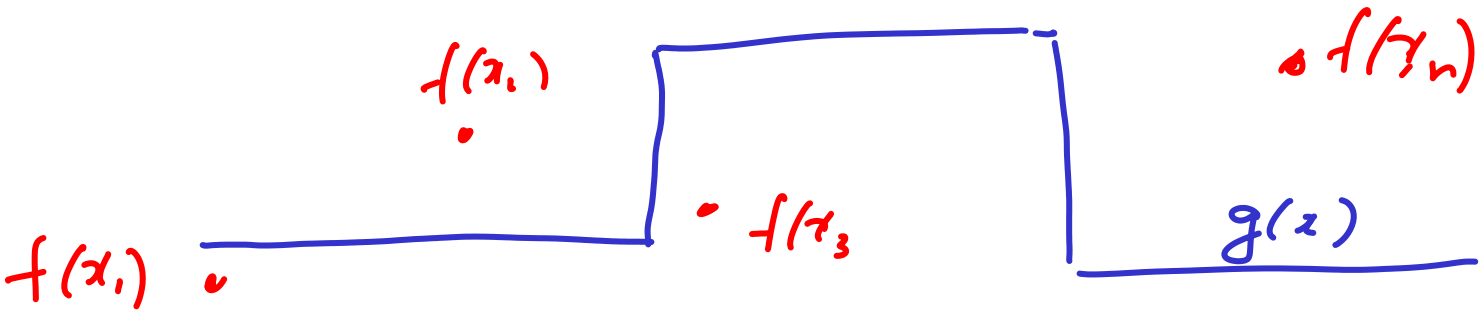
Lecture 18 COL 702 Sept 28

Function compression

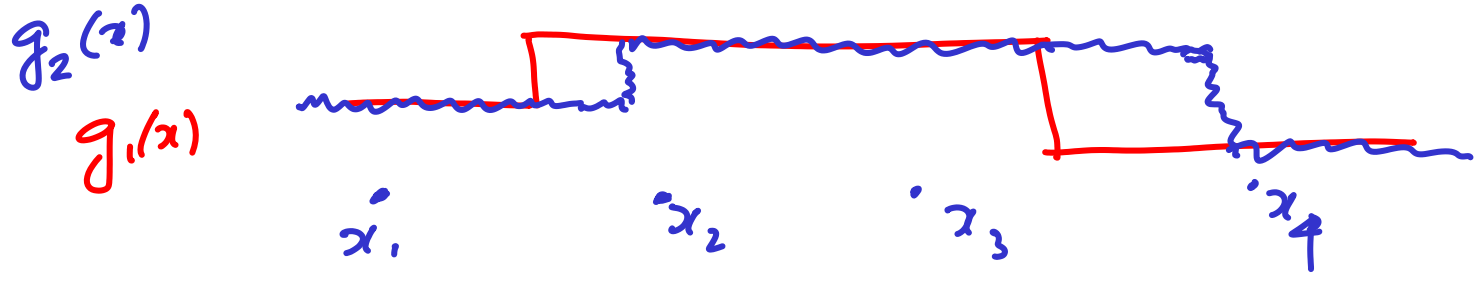


x_1 x_2 x_3 x_n

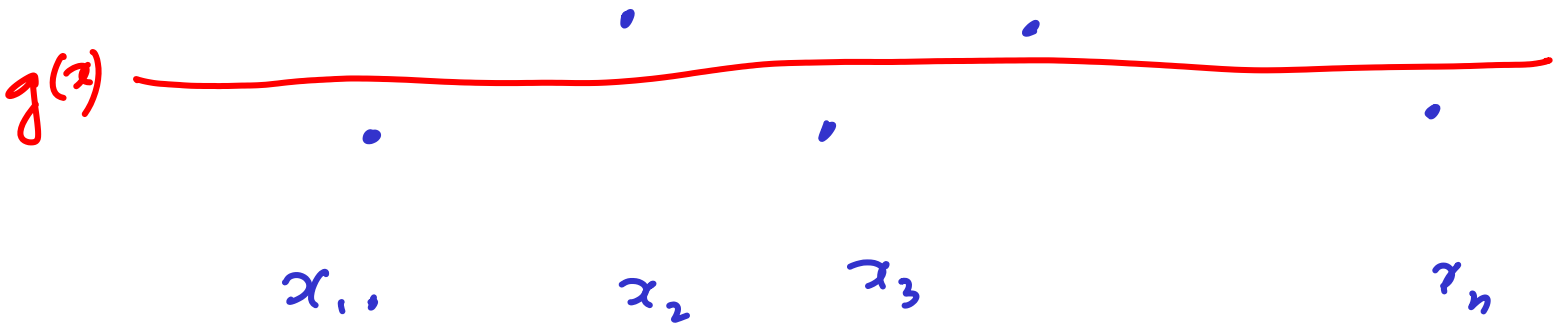
$$\Delta(f(x), g(x)) = \sum_{i=1}^{n} \left[g(x_i) - f(x_i) \right]^2$$

Minimize this for all choices of $g(x)$ where $g(x)$ is a k step function

Observations: $g(x)$ can be restricted to functions that have steps at some of the given values $\{x_1, x_2 \ldots x_n\}$



Observation: If $g(x)$ is a constant function, i.e. $k = 1$,



then $g(x) = \frac{1}{n} \sum_{i=1}^{n} f(x_i)$ (1-mean)

Let g_k^* is the best possible k-step function

g_1^* is easy to compute

g_2^* is defined by the critical point x_i that separates the two steps

Try all possibilities of x_i

Computation-time: $O(n^2)$ since for each choice of x_i, we can compute the mean in $O(n)$ time

Let $g_r^*(i,j)$ denote the best possible r-step function to approximate

$$[f_i \; f_{i+1} \; \cdots \; f_j]$$

We are interested in $g_k^*(1,n)$

Then write a recurrence for $g_r^*(i,j)$

Complete the recursive defns and analysis. Compare with the 2dim table approach

2nd approach

For every x_i find the best $g_{k-1}^*(1,i-1)$ and the remaining $x_{i+1} \cdots x_n$ can be approximated using the mean

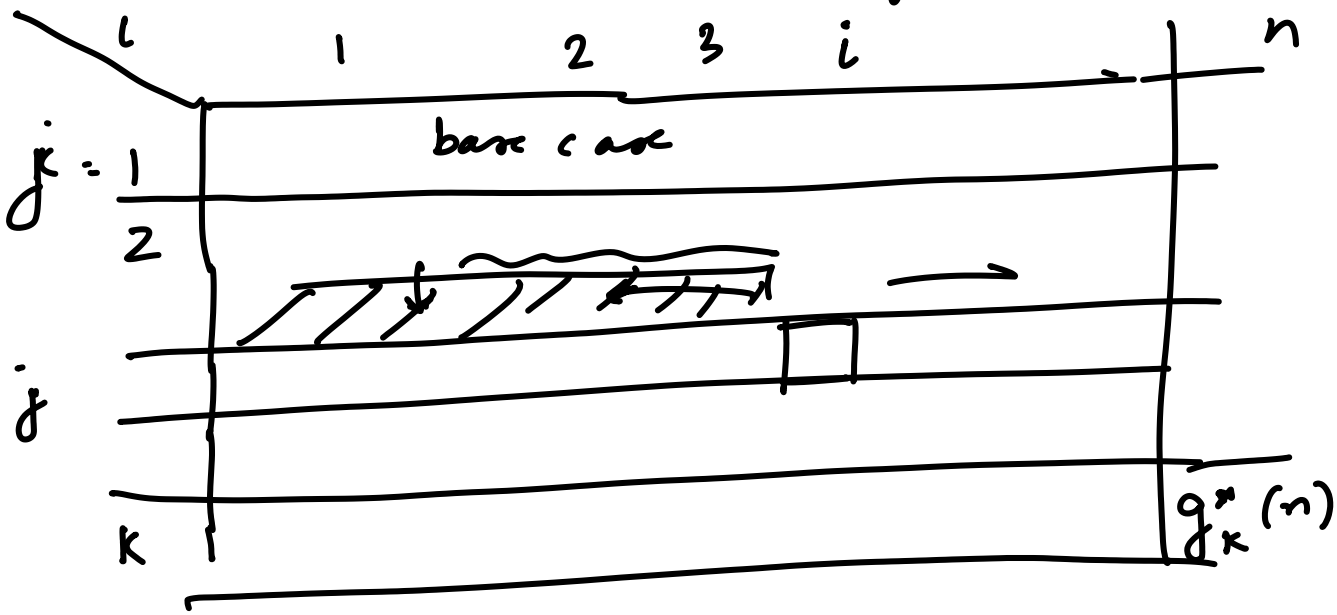
Choose the best x_i

$$g_k^*(1,i) = \min_{1 \le j \le i} \left\{ g_{k-1}^*(1,j) + \mu(j+1,i) \right\}$$

$\mu(j+1,i)$: optimal error given by mean

(Rewrite this by using $g_k^*(i)$ and ignore the fixed parameter 1)

What is the tabular form



To fill up the i^{th} entry of the j^{th} row we consult $i-1$ entries of $j-1$ row and mean of $i-j$ entries

$$\sum_{j=1}^{i-1} (i-j) \sim O(i^2)$$

By precomputing the $g_k(i,j)$ for all i,j, we can look up the corresponding mean to compute a table entry

Then filling up an entry $O(i^2)$

So for row j, the total cost is

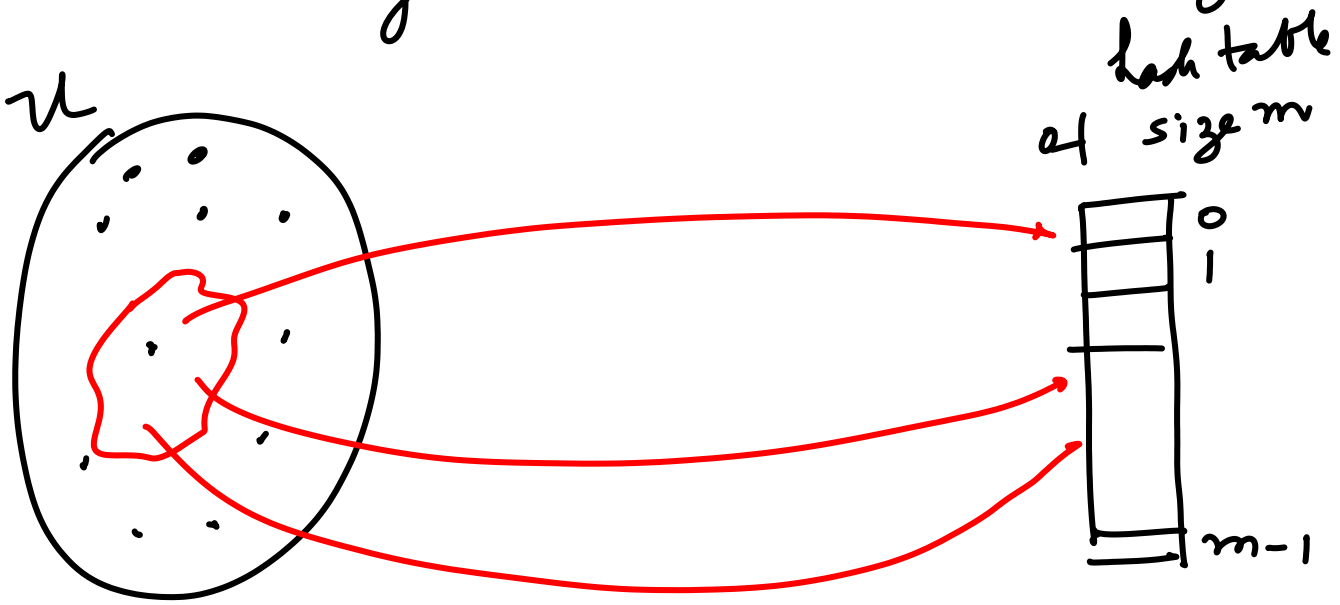
$$\sum_{i=1}^{n} O(i) \sim O(n^2)$$

$\Rightarrow$ Entire table can be computed in

$$O(n^2 k)$$

+ cost of computing all $\mu(i,j)$

Hashing as used searching



N possible key values

hash function

$h : U \to \{0, 1, \ldots m-1\}$

An instance of hashing involves
n keys from U $n \ll N$
and computing the hash values to
store them in the table T