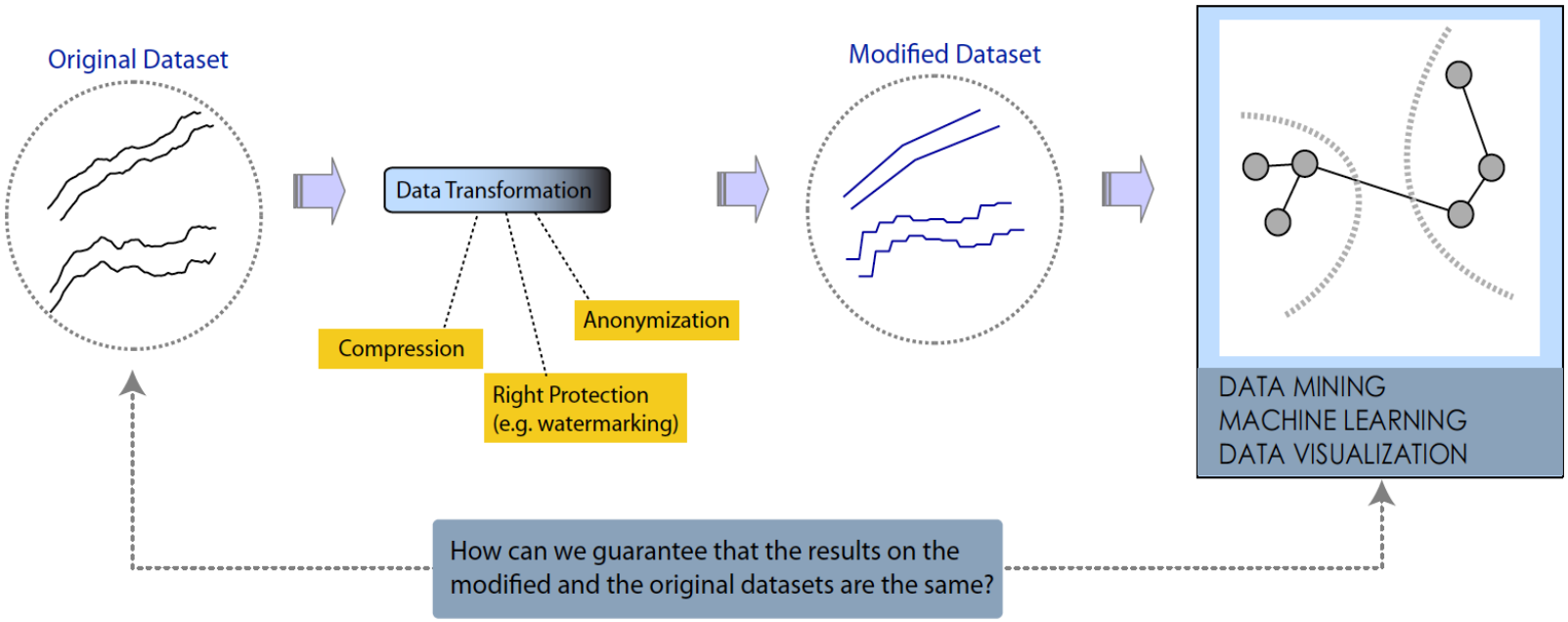
Optimal distance estimation on compressed data (the data mining perspective)



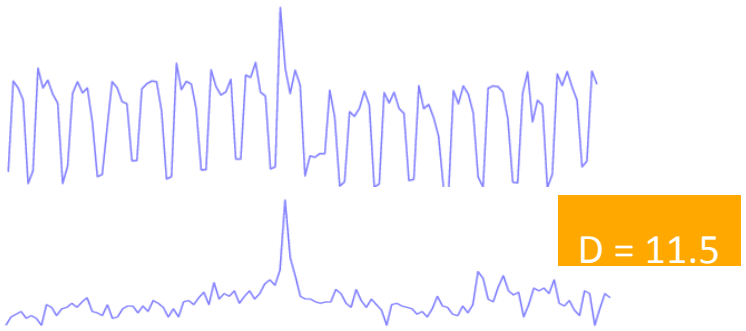
Nick Freris
NYU, Abu Dhabi

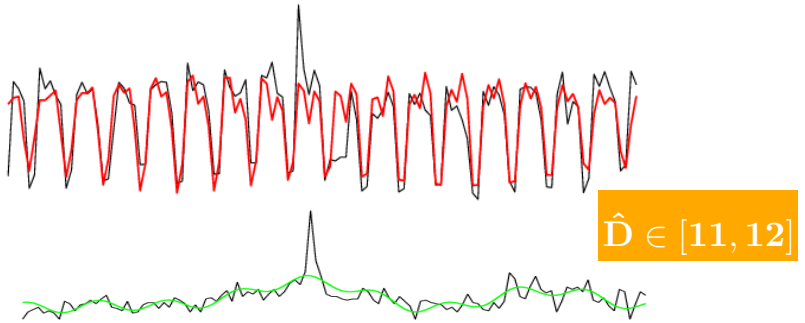
EMBOPT '14, IMT-Lucca
Sep. 9, 2014

NEW YORK UNIVERSITY
ABU DHABI

Optimal distance estimation

This work presents optimal estimation of Euclidean distance in the compressed domain. This result cannot be further improved





- Our approach is applicable on any orthonormal data compression basis (Fourier, Wavelets, Chebyshev, PCA, etc.)
- Our method allows up to
  - 57% better distance estimation
  - 80% less computation effort
  - 128 : 1 compression efficiency

Motivation/Benefits

Time-series data are customarily compressed in order to:

- Save storage space
- Reduce transmission bandwidth
- Achieve faster processing / data analysis
- Remove noise

Distance estimation has various data analytics applications

- Clustering / Classification
- Anomaly detection
- **Similarity search**

Now we can do all this very efficiently directly on the compressed data!

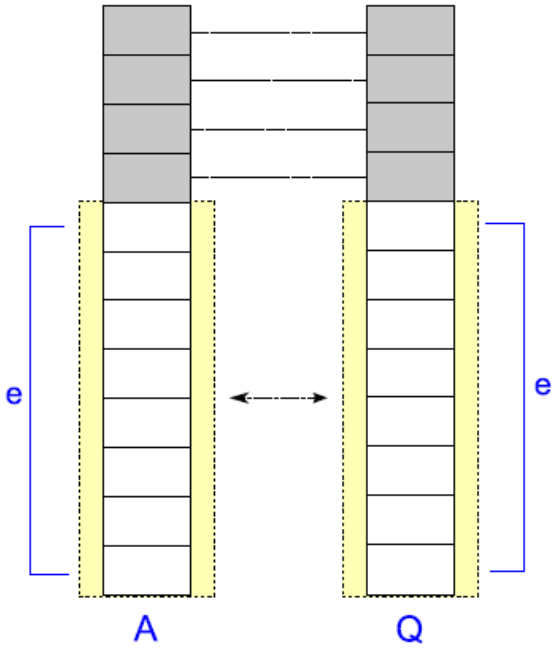
Previous work vs current approach

First coeffs + error

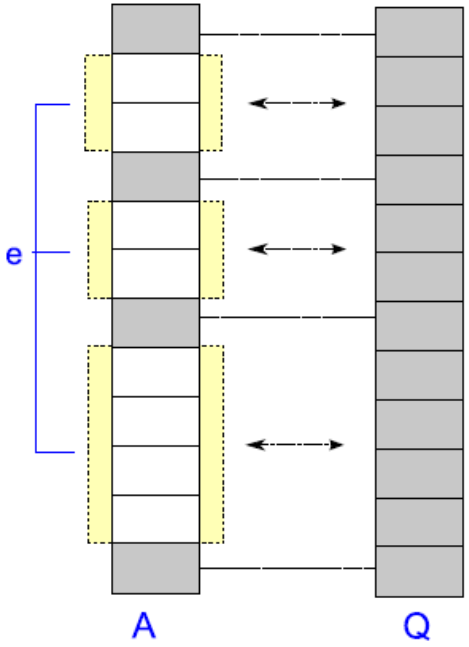
Best coeffs + error

Best coeffs + error + constraints → Optimal (our method)

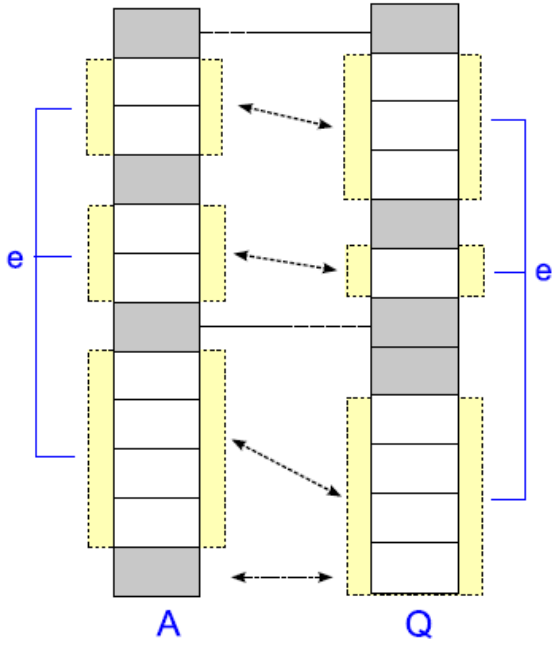
Approach 1



Approach 2:
(only one compressed)



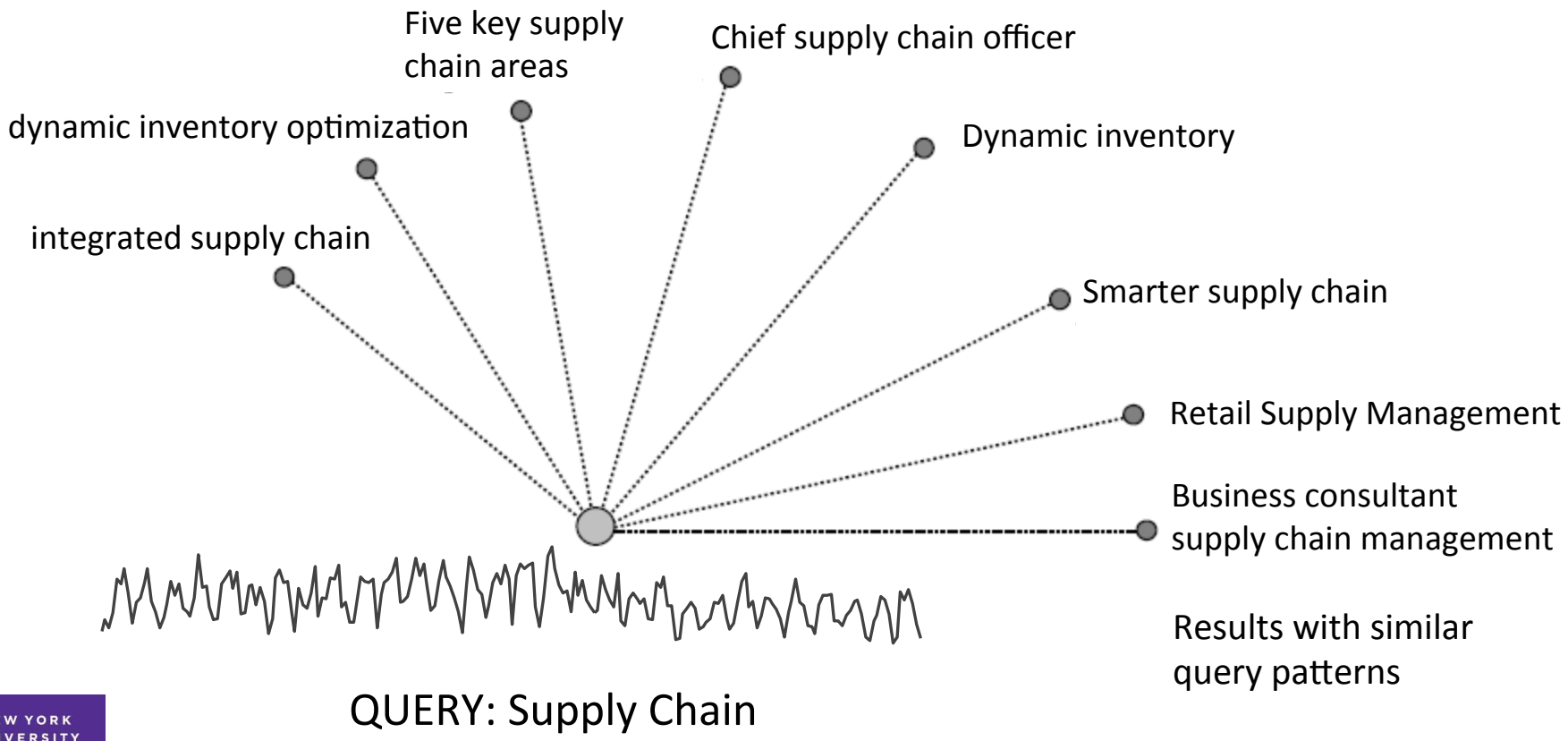
Our approach



applications

Similarity Search/Grouping/Clustering

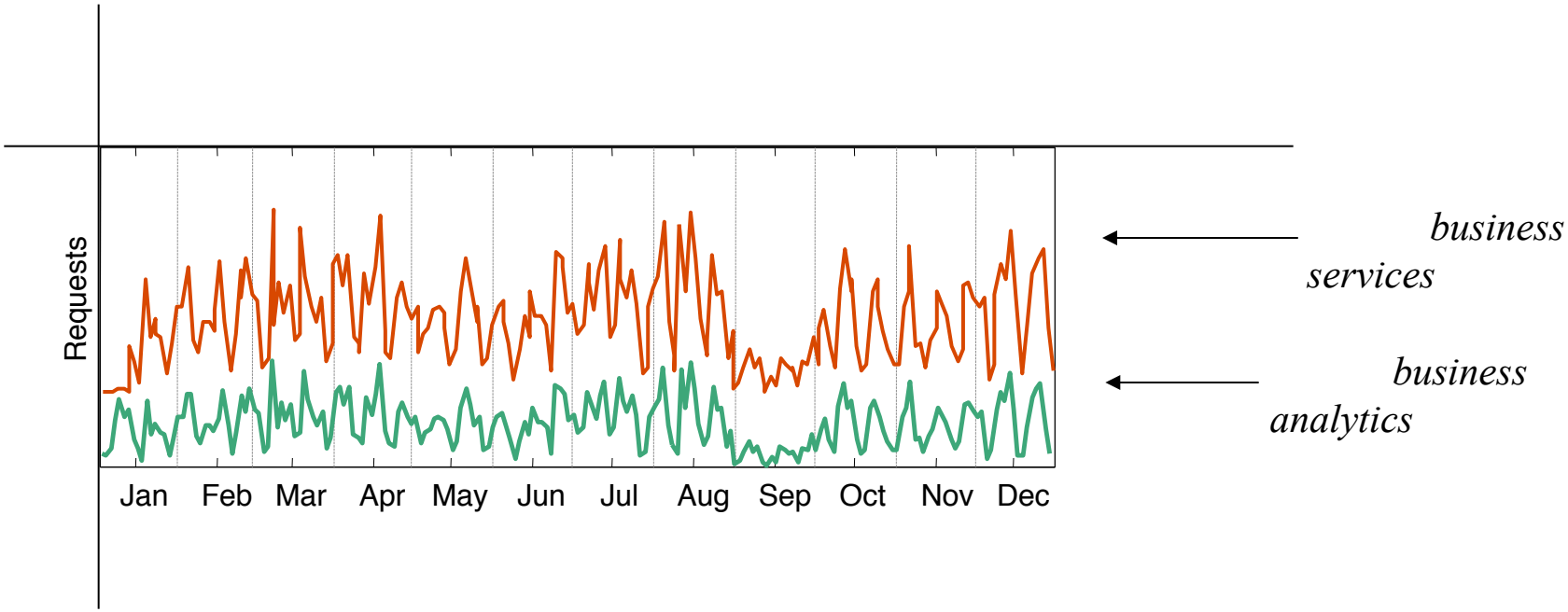
find semantically similar searches, deduce associations



applications

Similarity Search/Grouping/Clustering

- Advertising/recommendations
- Buying keywords that are cheaper but exhibit same demand pattern



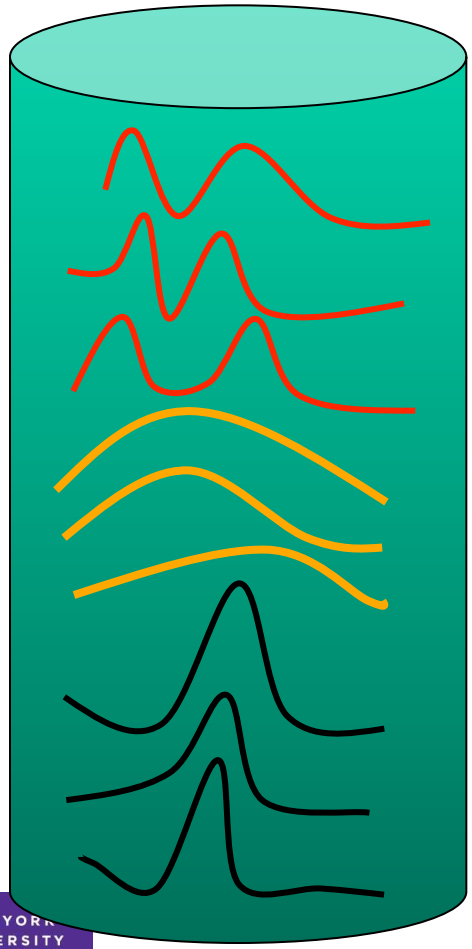
underlying operation: k-NN search

K-Nearest Neighbor (k-NN) similarity search

Issues that arise:

- How to compress data?
- How to speedup search
  - We have to estimate tight bounds on the distance metric using just the compressed representation

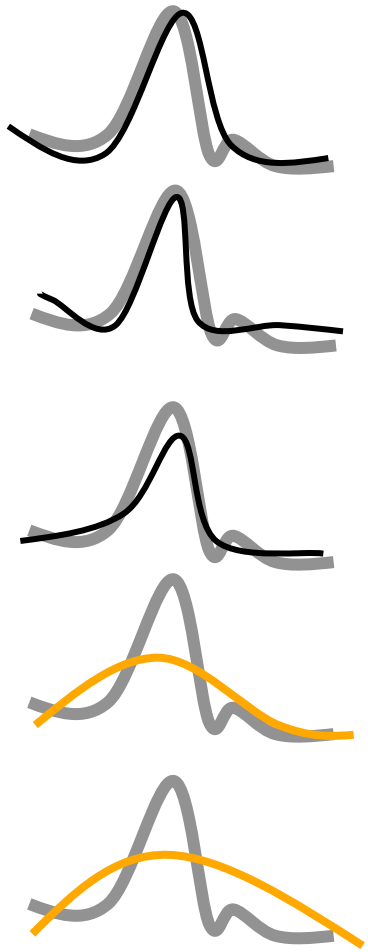
similarity search problem



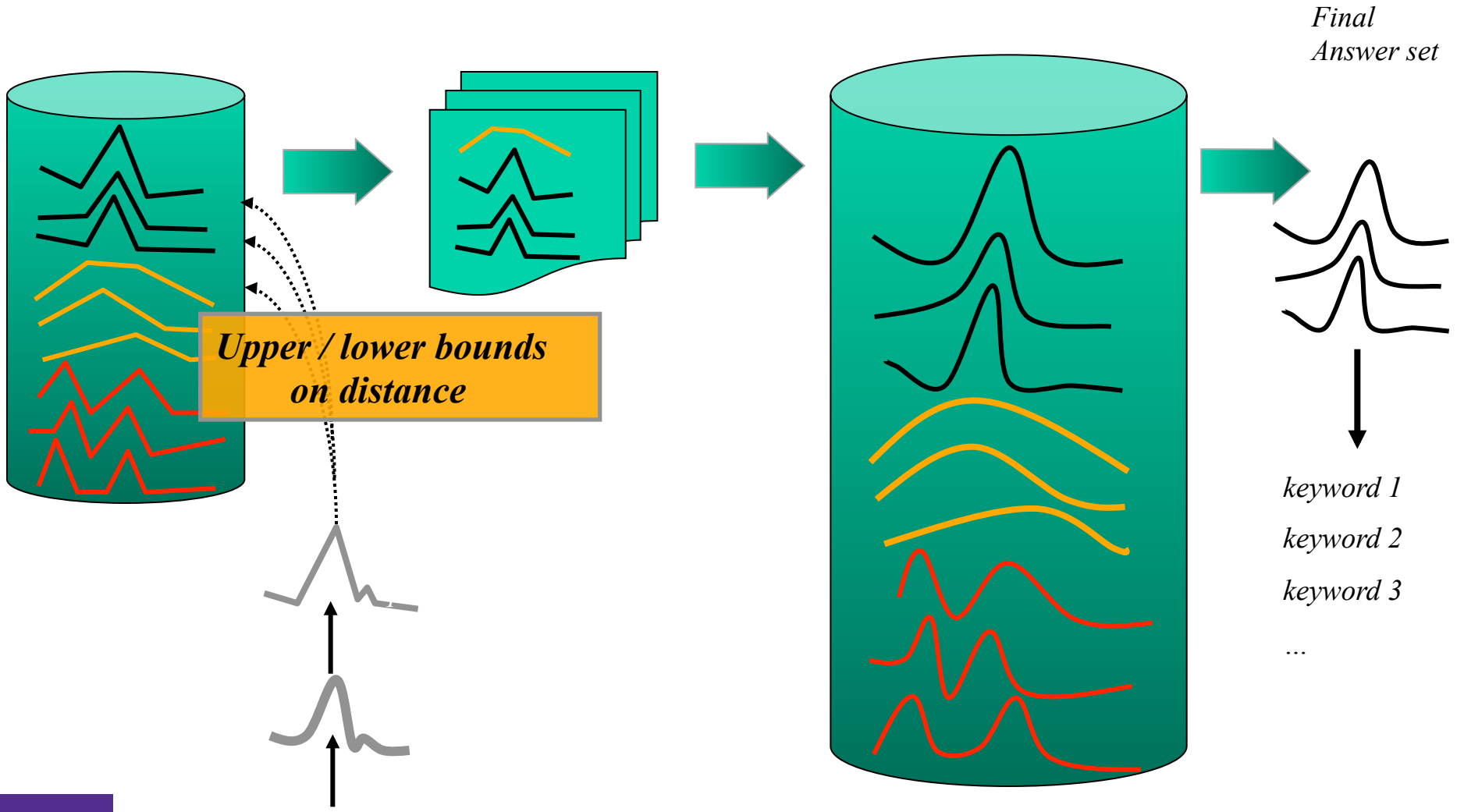
Linear Scan:

Objective: Compare the query with all sequences in DB and return the k most similar sequences to the query.

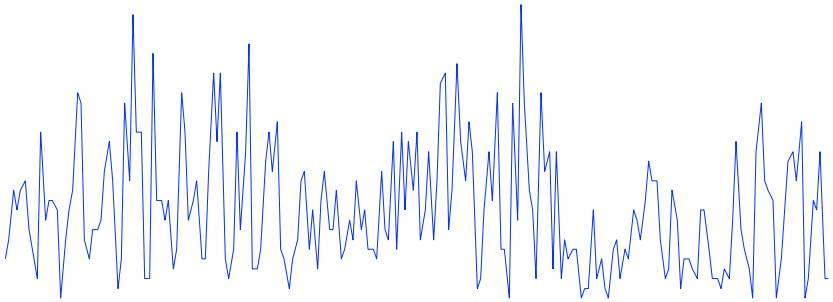
Distance



speeding up similarity search



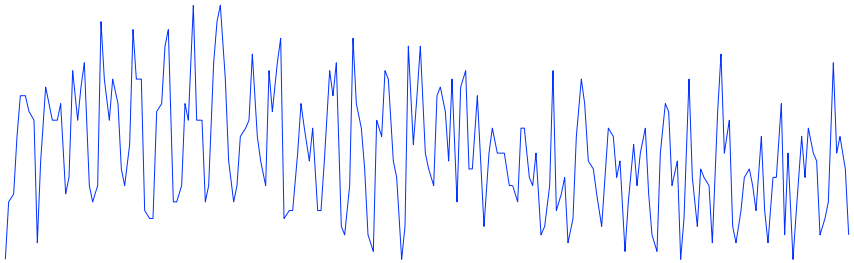
compressing weblog data



Query: "analytics and optimization"

*The data are highly **periodic**, so we can use Fourier decomposition.*

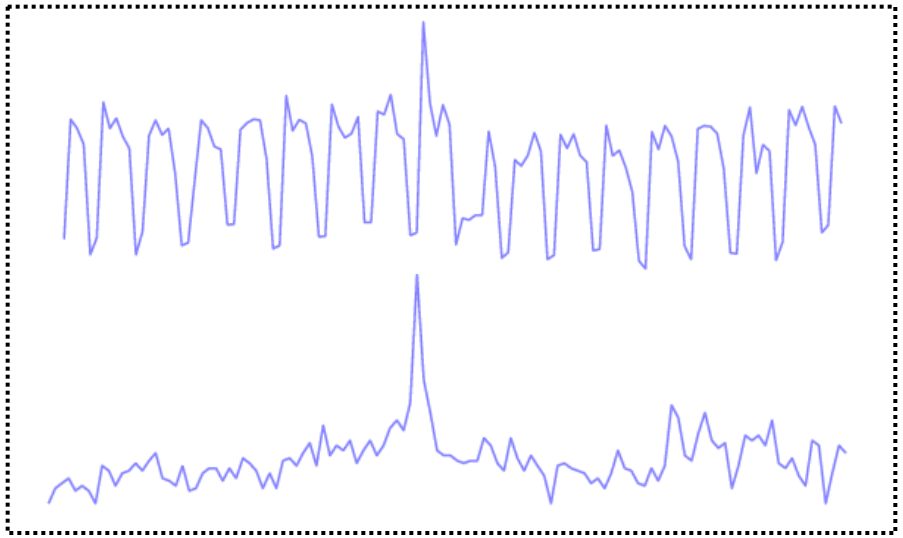
*Instead of using the first Fourier coefficients we can use the **best** ones instead.*



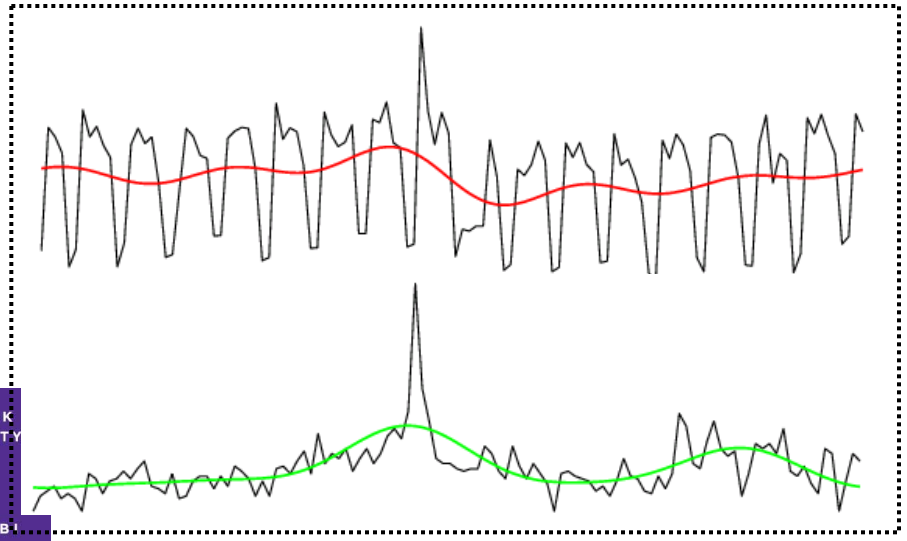
Query: "consulting services"

similarity Search

- Approximate the Euclidean distance using DFT



11.5517



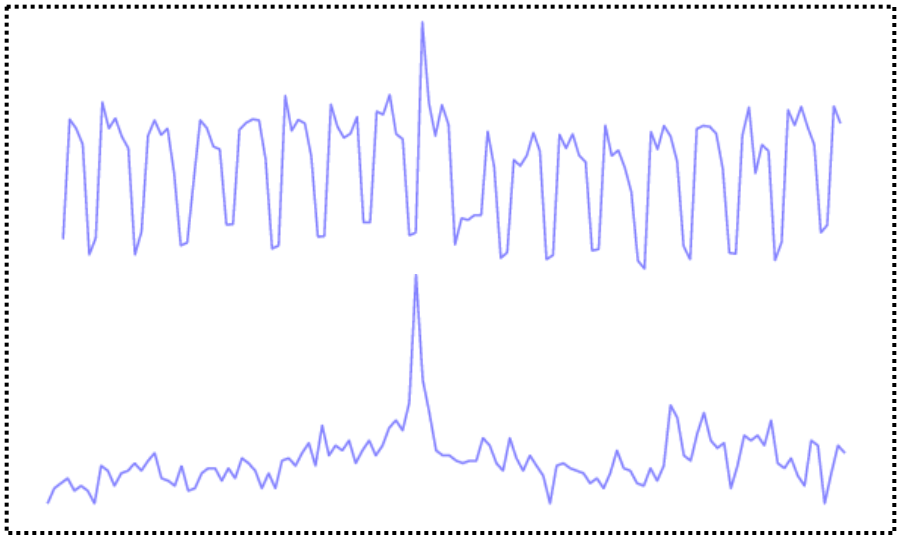
7.9234

$x(n)$	$X(f)$
0.4326	65.0630
2.0981	0.4721 +20.2755i
1.9728	4.6455 - 0.7204i
1.6851	-11.6083 - 5.8807i
2.8316	1.0588 - 3.9980i
1.6407	0.3014 + 3.4492i
0.4515	-1.3769 + 3.3947i
0.4892	0.5637 + 3.4632i
0.1619	-2.7711 + 0.3124i
0.0128	-3.6572 - 1.4000i
0.1739	0.1078 - 6.0011i
0.5518	-1.9892 + 2.8143i
0.0365	-2.6120 + 3.4182i
2.1467	-3.9104 - 0.4136i
2.0103	-1.2362 + 3.5155i
2.1243	-2.2397 - 0.6038i
3.1910	-2.7173
3.2503	-2.2397 + 0.6038i
3.1547	-1.2362 - 3.5155i
2.3223	-3.9104 + 0.4136i
2.6167	-2.6120 - 3.4182i
1.2805	-1.9892 - 2.8143i
1.9949	0.1078 + 6.0011i
3.6184	-3.6572 + 1.4000i
2.9266	-2.7711 - 0.3124i
3.7846	0.5637 - 3.4632i
5.0386	-1.3769 - 3.3947i
3.4449	0.3014 - 3.4492i
2.0039	1.0588 + 3.9980i
2.5751	-11.6083 + 5.8807i
2.1752	4.6455 + 0.7204i
2.8652	0.4721 -20.2755i

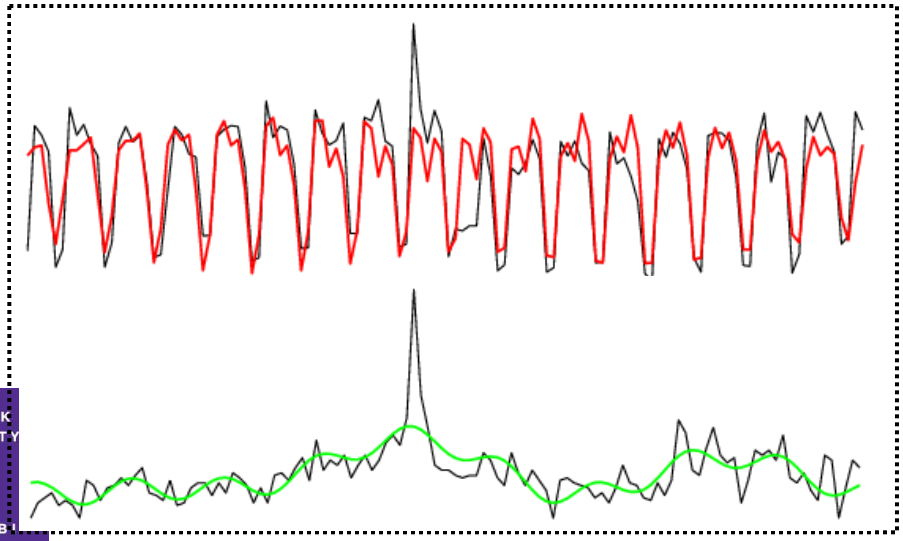
NEW YORK UNIVERSITY ABU DHABI

similarity Search

- Approximate the Euclidean distance using DFT



11.5517



11.1624

$x(n)$	$X(f)$
0.4326	65.0630
2.0981	0.4721 +20.2755i
1.9728	4.6455 - 0.7204i
1.6851	-11.6083 - 5.8807i
2.8316	1.0588 - 3.9980i
1.6407	0.3014 + 3.4492i
0.4515	-1.3769 + 3.3947i
0.4892	0.5637 + 3.4632i
0.1619	-2.7711 + 0.3124i
0.0128	-3.6572 - 1.4000i
0.1739	0.1078 - 6.0011i
0.5518	-1.9892 + 2.8143i
0.0365	-2.6120 + 3.4182i
2.1467	-3.9104 - 0.4136i
2.0103	-1.2362 + 3.5155i
2.1243	-2.2397 - 0.6038i
3.1910	-2.7173
3.2503	-2.2397 + 0.6038i
3.1547	-1.2362 - 3.5155i
2.3223	-3.9104 + 0.4136i
2.6167	-2.6120 - 3.4182i
1.2805	-1.9892 - 2.8143i
1.9949	0.1078 + 6.0011i
3.6184	-3.6572 + 1.4000i
2.9266	-2.7711 - 0.3124i
3.7846	0.5637 - 3.4632i
5.0386	-1.3769 - 3.3947i
3.4449	0.3014 - 3.4492i
2.0039	1.0588 + 3.9980i
2.5751	-11.6083 + 5.8807i
2.1752	4.6455 + 0.7204i
2.8652	0.4721 -20.2755i

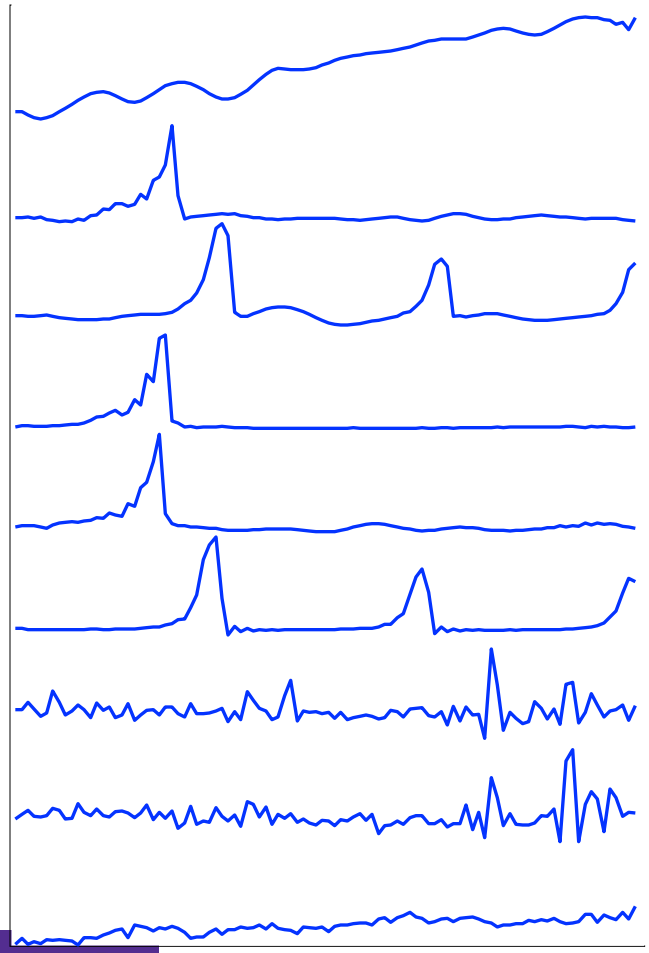
NEW YORK UNIVERSITY ABU DHABI

objective

Calculate the tightest possible upper/lower bounds using the coefficients with the highest energy

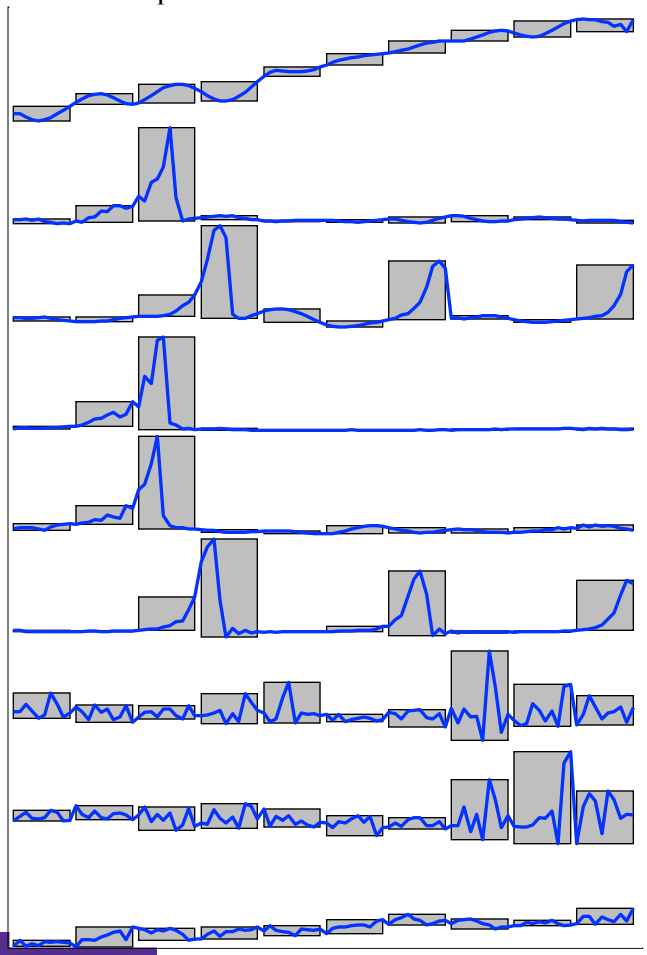
This will result in better pruning of the search space -> faster search

Lower/Upper Bounds

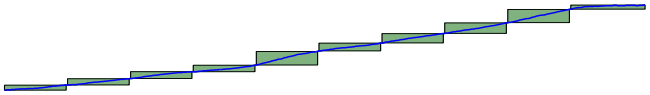


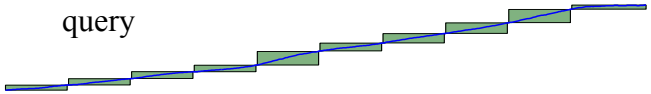
Lower/Upper Bounds

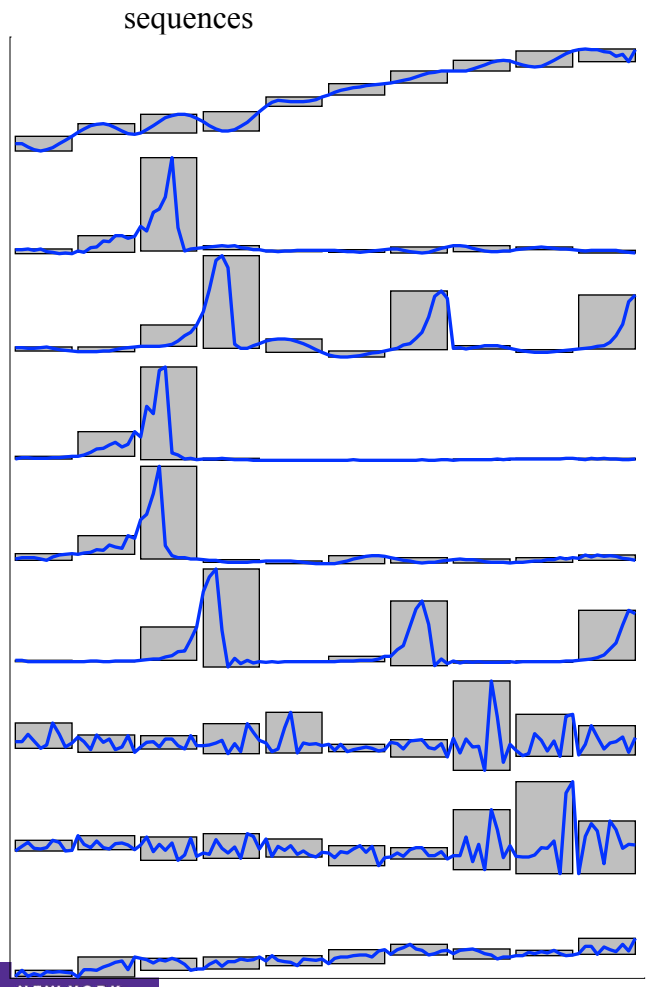
sequences



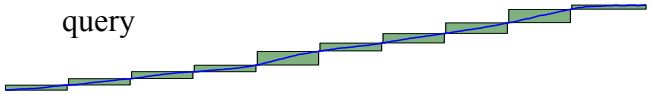
query

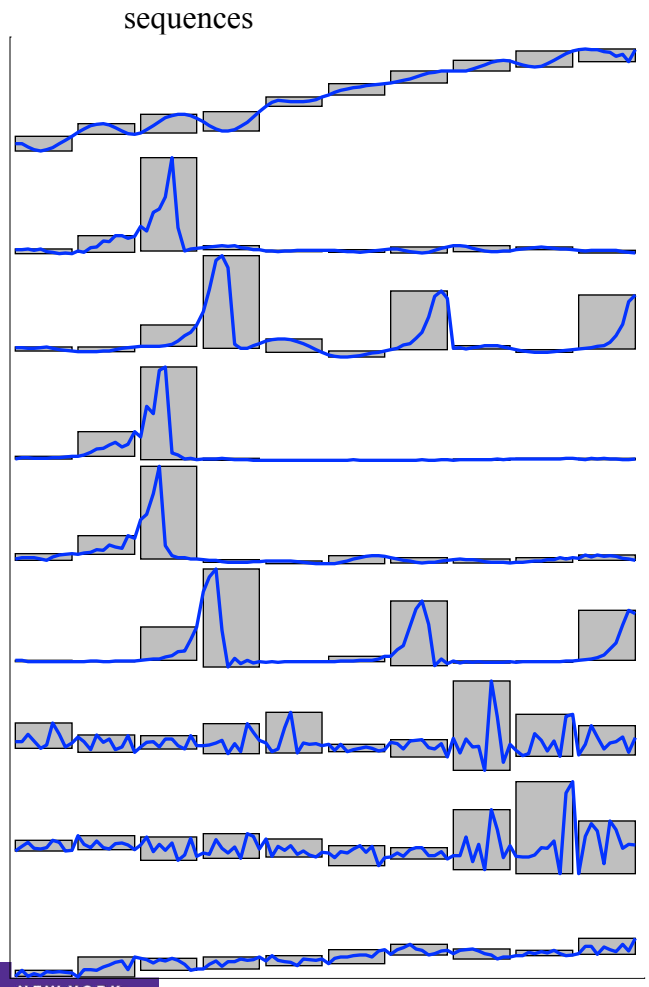






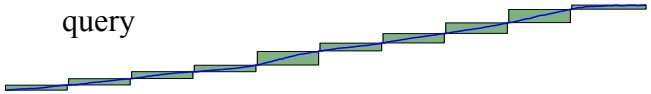
Lower Bounds	Upper Bounds	True Distance
4.6	126.1	46.7
37.9	259.7	108.8
19.5	291.3	113.5
72.1	198.9	104.5
67.1	208.0	119.4
78.0	190.4	120.0
70.9	187.1	111.6
63.7	201.0	119.0
1.4	46.6	17.2

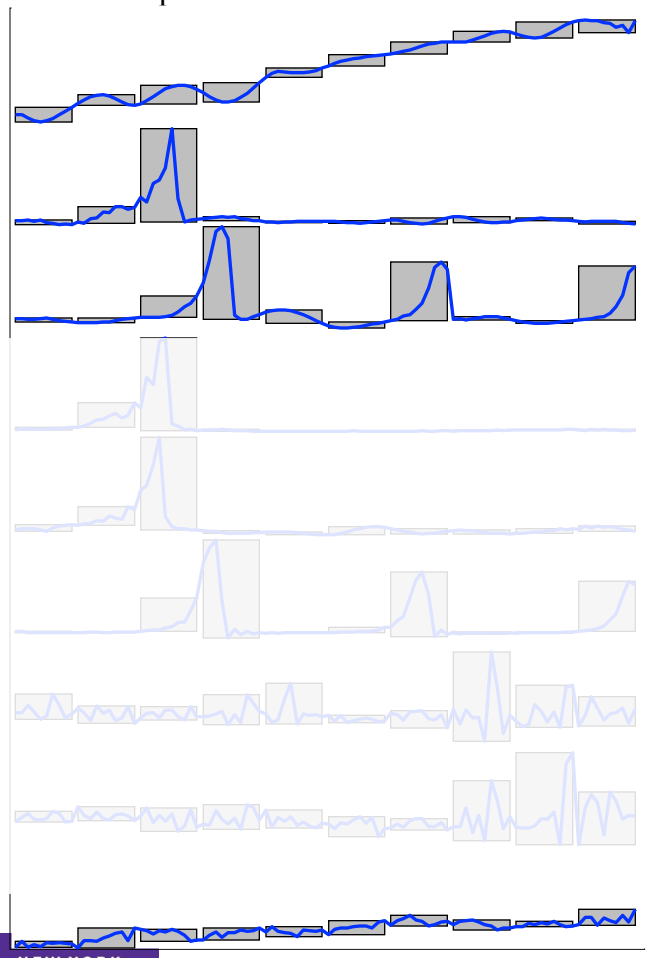




Lower Bounds	Upper Bounds	True Distance
4.6	126.1	46.7
37.9	259.7	108.8
19.5	291.3	113.5
72.1	198.9	104.5
67.1	208.	
78.0	190.	
70.9	187.	
63.7	201.0	119.0
1.4	46.6	17.2

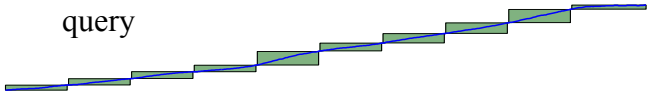
Anything with $LB > UB_{min}$ can be removed from examination

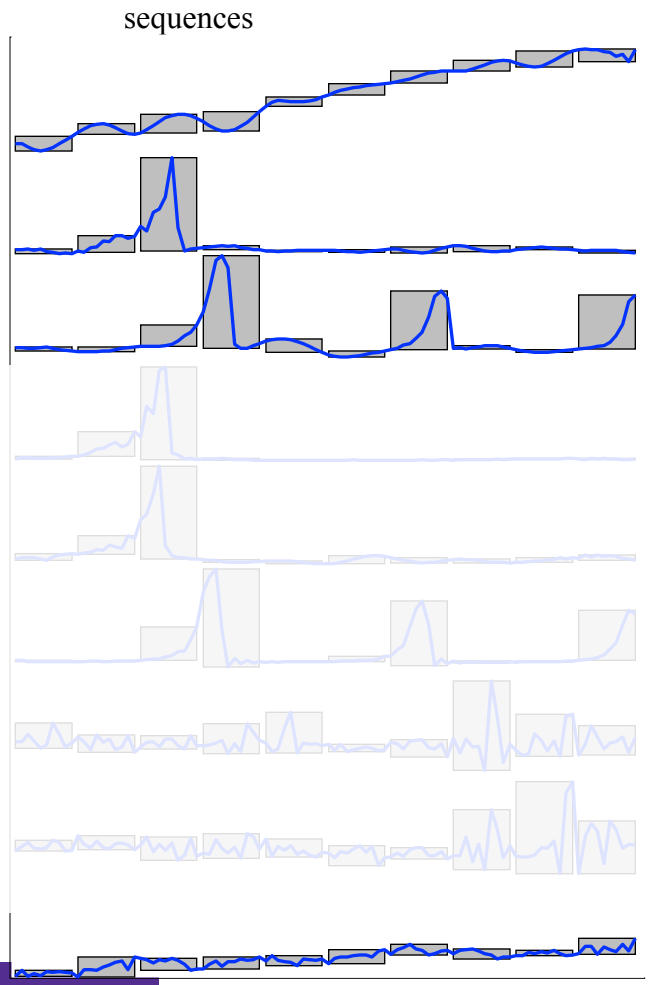




Lower Bounds	Upper Bounds	True Distance
4.6	126.1	46.7
37.9	259.7	108.8
19.5	291.3	113.5
72.1	198.9	104.5
67.1	208.0	
78.0	190.1	
70.9	187.1	
63.7	201.0	119.0
1.4	46.6	17.2

Anything with
$LB > UB_{min}$
can be removed
from examination



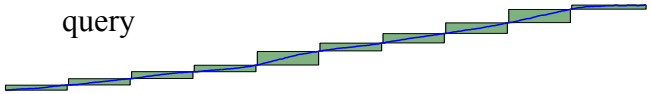


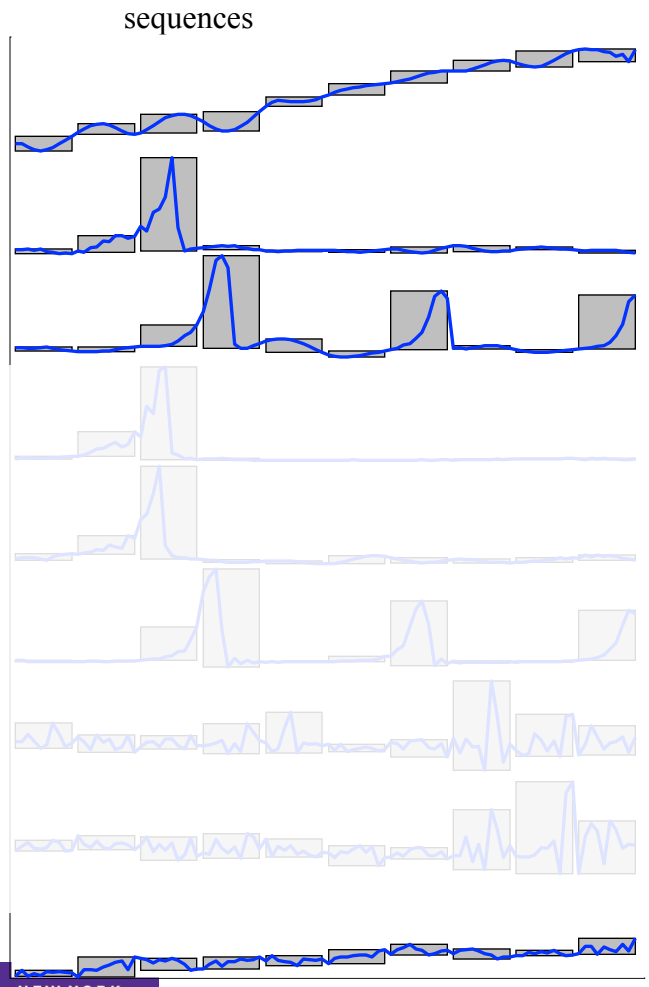
Lower Bounds	Upper Bounds	True Distance
4.6	126.1	46.7
37.9	259.7	108.8
19.5	291.3	113.5
72.1	198.9	104.5
67.1	208.[illegible]	[illegible]
78.0	190.[illegible]	[illegible]
70.9	187.[illegible]	[illegible]
63.7	201.0	[illegible]
1.4	46.6	17.2

Now we start fetching from disk in the order suggested by the LB

BestSoFar

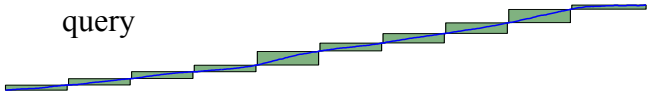
NEW YORK UNIVERSITY ABU DHABI

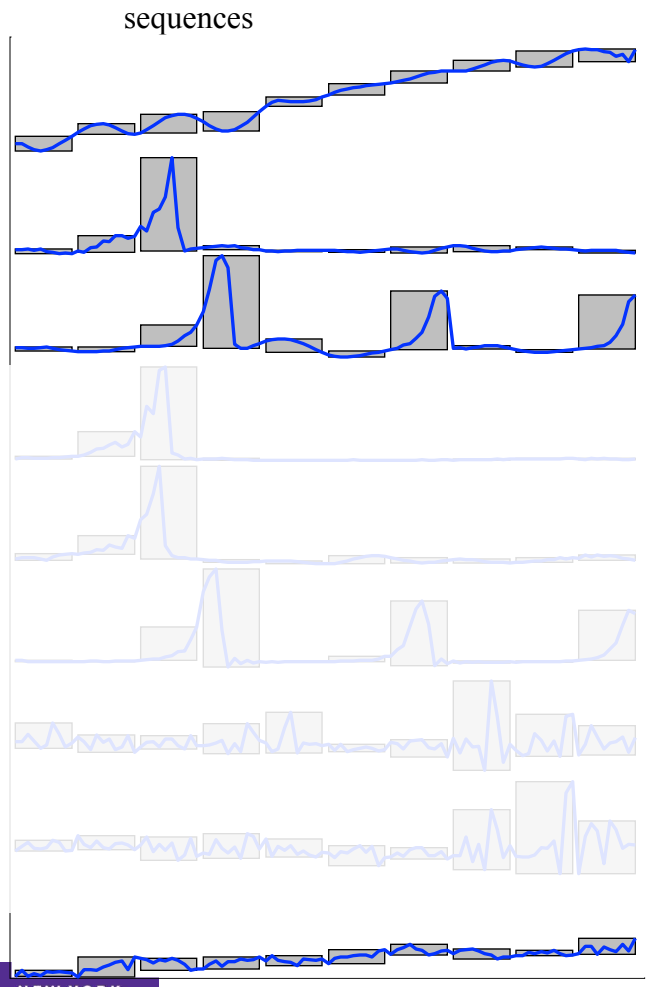




Lower Bounds	Upper Bounds	True Distance
4.6	126.1	46.7
37.9	259.7	108.8
19.5	291.3	113.5
72.1	198.9	104.5
67.1	208.0	119.4
78.0	190.4	120.0
70.9	187.1	111.6
63.7	201.0	BestSoFar
1.4	46.6	17.2

NEW YORK UNIVERSITY ABU DHABI





Lower Bounds

True Distance

Lower Bounds	True Distance
4.6	46.7
37.9	108.8
19.5	113.5
72.1	104.5
67.1	119.4
78.0	120.0
70.9	111.6
63.7	BestSoFar
1.4	17.2

Now we can stop.
$LB_{current}$ > BestSoFar
(19.5 > 17.2)

x	X	Q	q
-1.7313	0.0000	0.0000	-1.3356
-0.7221	-12.0861 + 4.4812i	-2.4756 + 1.7973i	0.6182
-1.2267	7.0708 - 3.3545i	-2.8455 - 0.8510i	-1.6135
-0.4194	0.8045 + 4.2567i	-5.5245 - 2.4452i	1.0842
-0.4194	0.2386 + 2.0592i	-2.4342 - 1.9897i	0.7981
-0.0158	-1.6003 + 6.7154i	-2.1082 + 1.6303i	0.2667
-0.8231	-2.6539 + 0.8595i	-3.6438 - 1.9424i	-1.2048
-1.2267	5.8845 + 3.7689i	-1.5989 - 0.1514i	0.7654
-0.3185	-2.0182 + 3.9356i	1.6186 - 2.3544i	0.6182
-0.3185	-3.9484 + 0.2234i	15.2057 - 6.0515i	-1.4500
-0.1167	-1.2440 + 2.4538i	-3.6746 + 0.3413i	-0.2892
-0.5203	-1.6268 + 1.0393i	-0.0532 + 0.6724i	0.6264
0.0851	-1.0458 + 4.0774i	-1.8331 - 0.6000i	0.2503
0.6906	-4.2436 + 0.5988i	0.1642 + 1.3907i	-1.0740
0.1861	-6.4020 - 0.9529i	-7.3632 - 5.7938i	0.7163
0.7915	-3.3663 - 1.0825i	-2.2362 - 1.7287i	0.7654
0.7915	-2.9264	-5.1339	-1.5073
1.2961	-3.3663 + 1.0825i	-2.2362 + 1.7287i	1.1987
2.3052	-6.4020 + 0.9529i	-7.3632 + 5.7938i	0.7735
1.9016	-4.2436 - 0.5988i	0.1642 - 1.3907i	0.1277
0.1861	-1.0458 - 4.0774i	-1.8331 + 0.6000i	-1.2865
-0.0158	-1.6268 - 1.0393i	-0.0532 - 0.6724i	0.6509
0.5897	-1.2440 - 2.4538i	-3.6746 - 0.3413i	0.5283
0.8924	-3.9484 - 0.2234i	15.2057 + 6.0515i	-1.0413
0.1861	-2.0182 - 3.9356i	1.6186 + 2.3544i	0.9207
-1.7313	5.8845 - 3.7689i	-1.5989 + 0.1514i	1.2150
-0.3185	-2.6539 - 0.8595i	-3.6438 + 1.9424i	0.4302
-0.9240	-1.6003 - 6.7154i	-2.1082 - 1.6303i	-1.2211
-0.5203	0.2386 - 2.0592i	-2.4342 + 1.9897i	1.0678
-0.4194	0.8045 - 4.2567i	-5.5245 + 2.4452i	1.0351
-0.3185	7.0708 + 3.3545i	-2.8455 + 0.8510i	-1.4337
2.2043	-12.0861 - 4.4812i	-2.4756 - 1.7973i	-1.0004

NEW YORK UNIVERSITY ABU DHABI

- Find best k coefficients of X (k=4)

Magnitude vector

||X||	X	Q	||Q||
0.0000	0.0000	0.0000	0
12.8901	-12.0861 + 4.4812i	-2.4756 + 1.7973i	3.0592
7.8261	7.0708 - 3.3545i	-2.8455 - 0.8510i	2.9700
4.3321	0.8045 + 4.2567i	-5.5245 - 2.4452i	6.0414
2.0729	0.2386 + 2.0592i	-2.4342 - 1.9897i	3.1439
6.9035	-1.6003 + 6.7154i	-2.1082 + 1.6303i	2.6650
2.7897	-2.6539 + 0.8595i	-3.6438 - 1.9424i	4.1292
6.9880	5.8845 + 3.7689i	-1.5989 - 0.1514i	1.6061
4.4229	-2.0182 + 3.9356i	1.6186 - 2.3544i	2.8571
3.9548	-3.9484 + 0.2234i	15.2057 - 6.0515i	16.3656
2.7512	-1.2440 + 2.4538i	-3.6746 + 0.3413i	3.6904
1.9304	-1.6268 + 1.0393i	-0.0532 + 0.6724i	0.6745
4.2094	-1.0458 + 4.0774i	-1.8331 - 0.6000i	1.9288
4.2856	-4.2436 + 0.5988i	0.1642 + 1.3907i	1.4004
6.4725	-6.4020 - 0.9529i	-7.3632 - 5.7938i	9.3694
3.5360	-3.3663 - 1.0825i	-2.2362 - 1.7287i	2.8265
2.9264	-2.9264	-5.1339	5.1339
3.5360	-3.3663 + 1.0825i	-2.2362 + 1.7287i	2.8265
6.4725	-6.4020 + 0.9529i	-7.3632 + 5.7938i	9.3694
4.2856	-4.2436 - 0.5988i	0.1642 - 1.3907i	1.4004
4.2094	-1.0458 - 4.0774i	-1.8331 + 0.6000i	1.9288
1.9304	-1.6268 - 1.0393i	-0.0532 - 0.6724i	0.6745
2.7512	-1.2440 - 2.4538i	-3.6746 - 0.3413i	3.6904
3.9548	-3.9484 - 0.2234i	15.2057 + 6.0515i	16.3656
4.4229	-2.0182 - 3.9356i	1.6186 + 2.3544i	2.8571
6.9880	5.8845 - 3.7689i	-1.5989 + 0.1514i	1.6061
2.7897	-2.6539 - 0.8595i	-3.6438 + 1.9424i	4.1292
6.9035	-1.6003 - 6.7154i	-2.1082 - 1.6303i	2.6650
2.0729	0.2386 - 2.0592i	-2.4342 + 1.9897i	3.1439
4.3321	0.8045 - 4.2567i	-5.5245 + 2.4452i	6.0414
7.8261	7.0708 + 3.3545i	-2.8455 + 0.8510i	2.9700
12.8901	-12.0861 - 4.4812i	-2.4756 - 1.7973i	3.0592

- Find best 4 coefficients of X

	X	Q	||Q||
0.0000	0.0000	0.0000	0
12.8901	-12.0861 + 4.4812i	-2.4756 + 1.7973i	3.0592
7.8261	7.0708 - 3.3545i	-2.8455 - 0.8510i	2.9700
4.3321	0.8045 + 4.2567i	-5.5245 - 2.4452i	6.0414
2.0729	0.2386 + 2.0592i	-2.4342 - 1.9897i	3.1439
6.9035	-1.6003 + 6.7154i	-2.1082 + 1.6303i	2.6650
2.7897	-2.6539 + 0.8595i	-3.6438 - 1.9424i	4.1292
6.9880	5.8845 + 3.7689i	-1.5989 - 0.1514i	1.6061
4.4229	-2.0182 + 3.9356i	1.6186 - 2.3544i	2.8571
3.9548	-3.9484 + 0.2234i	15.2057 - 6.0515i	16.3656
2.7512	-1.2440 + 2.4538i	-3.6746 + 0.3413i	3.6904
1.9304	-1.6268 + 1.0393i	-0.0532 + 0.6724i	0.6745
4.2094	-1.0458 + 4.0774i	-1.8331 - 0.6000i	1.9288
4.2856	-4.2436 + 0.5988i	0.1642 + 1.3907i	1.4004
6.4725	-6.4020 - 0.9529i	-7.3632 - 5.7938i	9.3694
3.5360	-3.3663 - 1.0825i	-2.2362 - 1.7287i	2.8265
2.9264	-2.9264	-5.1339	5.1339
3.5360	-3.3663 + 1.0825i	-2.2362 + 1.7287i	2.8265
6.4725	-6.4020 + 0.9529i	-7.3632 + 5.7938i	9.3694
4.2856	-4.2436 - 0.5988i	0.1642 - 1.3907i	1.4004
4.2094	-1.0458 - 4.0774i	-1.8331 + 0.6000i	1.9288
1.9304	-1.6268 - 1.0393i	-0.0532 - 0.6724i	0.6745
2.7512	-1.2440 - 2.4538i	-3.6746 - 0.3413i	3.6904
3.9548	-3.9484 - 0.2234i	15.2057 + 6.0515i	16.3656
4.4229	-2.0182 - 3.9356i	1.6186 + 2.3544i	2.8571
6.9880	5.8845 - 3.7689i	-1.5989 + 0.1514i	1.6061
2.7897	-2.6539 - 0.8595i	-3.6438 + 1.9424i	4.1292
6.9035	-1.6003 - 6.7154i	-2.1082 - 1.6303i	2.6650
2.0729	0.2386 - 2.0592i	-2.4342 + 1.9897i	3.1439
4.3321	0.8045 - 4.2567i	-5.5245 + 2.4452i	6.0414
7.8261	7.0708 + 3.3545i	-2.8455 + 0.8510i	2.9700
12.8901	-12.0861 - 4.4812i	-2.4756 - 1.7973i	3.0592

NEW YORK UNIVERSITY ABU DHABI

- Identify smallest magnitude (→ *power*)

minPower
6.9035 →

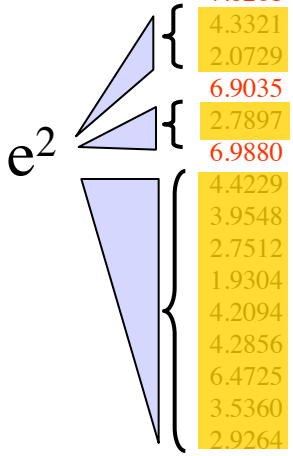
	X	Q	‖Q‖
0.0000	0.0000	0.0000	0
12.8901	-12.0861 + 4.4812i	-2.4756 + 1.7973i	3.0592
7.8261	7.0708 - 3.3545i	-2.8455 - 0.8510i	2.9700
4.3321	0.8045 + 4.2567i	-5.5245 - 2.4452i	6.0414
2.0729	0.2386 + 2.0592i	-2.4342 - 1.9897i	3.1439
6.9035	-1.6003 + 6.7154i	-2.1082 + 1.6303i	2.6650
2.7897	-2.6539 + 0.8595i	-3.6438 - 1.9424i	4.1292
6.9880	5.8845 + 3.7689i	-1.5989 - 0.1514i	1.6061
4.4229	-2.0182 + 3.9356i	1.6186 - 2.3544i	2.8571
3.9548	-3.9484 + 0.2234i	15.2057 - 6.0515i	16.3656
2.7512	-1.2440 + 2.4538i	-3.6746 + 0.3413i	3.6904
1.9304	-1.6268 + 1.0393i	-0.0532 + 0.6724i	0.6745
4.2094	-1.0458 + 4.0774i	-1.8331 - 0.6000i	1.9288
4.2856	-4.2436 + 0.5988i	0.1642 + 1.3907i	1.4004
6.4725	-6.4020 - 0.9529i	-7.3632 - 5.7938i	9.3694
3.5360	-3.3663 - 1.0825i	-2.2362 - 1.7287i	2.8265
2.9264	-2.9264	-5.1339	5.1339
3.5360	-3.3663 + 1.0825i	-2.2362 + 1.7287i	2.8265
6.4725	-6.4020 + 0.9529i	-7.3632 + 5.7938i	9.3694
4.2856	-4.2436 - 0.5988i	0.1642 - 1.3907i	1.4004
4.2094	-1.0458 - 4.0774i	-1.8331 + 0.6000i	1.9288
1.9304	-1.6268 - 1.0393i	-0.0532 - 0.6724i	0.6745
2.7512	-1.2440 - 2.4538i	-3.6746 - 0.3413i	3.6904
3.9548	-3.9484 - 0.2234i	15.2057 + 6.0515i	16.3656
4.4229	-2.0182 - 3.9356i	1.6186 + 2.3544i	2.8571
6.9880	5.8845 - 3.7689i	-1.5989 + 0.1514i	1.6061
2.7897	-2.6539 - 0.8595i	-3.6438 + 1.9424i	4.1292
6.9035	-1.6003 - 6.7154i	-2.1082 - 1.6303i	2.6650
2.0729	0.2386 - 2.0592i	-2.4342 + 1.9897i	3.1439
4.3321	0.8045 - 4.2567i	-5.5245 + 2.4452i	6.0414
7.8261	7.0708 + 3.3545i	-2.8455 + 0.8510i	2.9700
12.8901	-12.0861 - 4.4812i	-2.4756 - 1.7973i	3.0592

- The remaining powers are less than minPower

minPower 6.9035

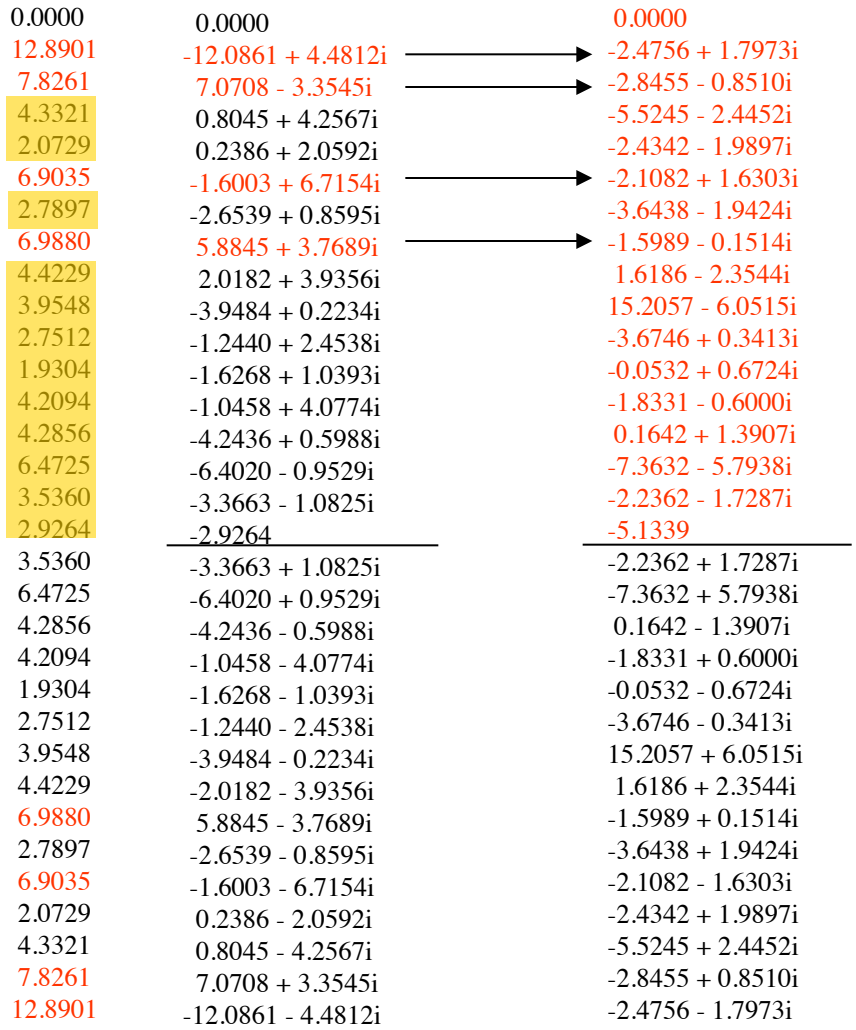
	X	Q	||Q||
0.0000	0.0000	0.0000	0
12.8901	-12.0861 + 4.4812i	-2.4756 + 1.7973i	3.0592
7.8261	7.0708 - 3.3545i	-2.8455 - 0.8510i	2.9700
4.3321	0.8045 + 4.2567i	-5.5245 - 2.4452i	6.0414
2.0729	0.2386 + 2.0592i	-2.4342 - 1.9897i	3.1439
6.9035	-1.6003 + 6.7154i	-2.1082 + 1.6303i	2.6650
2.7897	-2.6539 + 0.8595i	-3.6438 - 1.9424i	4.1292
6.9880	5.8845 + 3.7689i	-1.5989 - 0.1514i	1.6061
4.4229	-2.0182 + 3.9356i	1.6186 - 2.3544i	2.8571
3.9548	-3.9484 + 0.2234i	15.2057 - 6.0515i	16.3656
2.7512	-1.2440 + 2.4538i	-3.6746 + 0.3413i	3.6904
1.9304	-1.6268 + 1.0393i	-0.0532 + 0.6724i	0.6745
4.2094	-1.0458 + 4.0774i	-1.8331 - 0.6000i	1.9288
4.2856	-4.2436 + 0.5988i	0.1642 + 1.3907i	1.4004
6.4725	-6.4020 - 0.9529i	-7.3632 - 5.7938i	9.3694
3.5360	-3.3663 - 1.0825i	-2.2362 - 1.7287i	2.8265
2.9264	-2.9264	-5.1339	5.1339
3.5360	-3.3663 + 1.0825i	-2.2362 + 1.7287i	2.8265
6.4725	-6.4020 + 0.9529i	-7.3632 + 5.7938i	9.3694
4.2856	-4.2436 - 0.5988i	0.1642 - 1.3907i	1.4004
4.2094	-1.0458 - 4.0774i	-1.8331 + 0.6000i	1.9288
1.9304	-1.6268 - 1.0393i	-0.0532 - 0.6724i	0.6745
2.7512	-1.2440 - 2.4538i	-3.6746 - 0.3413i	3.6904
3.9548	-3.9484 - 0.2234i	15.2057 + 6.0515i	16.3656
4.4229	-2.0182 - 3.9356i	1.6186 + 2.3544i	2.8571
6.9880	5.8845 - 3.7689i	-1.5989 + 0.1514i	1.6061
2.7897	-2.6539 - 0.8595i	-3.6438 + 1.9424i	4.1292
6.9035	-1.6003 - 6.7154i	-2.1082 - 1.6303i	2.6650
2.0729	0.2386 - 2.0592i	-2.4342 + 1.9897i	3.1439
4.3321	0.8045 - 4.2567i	-5.5245 + 2.4452i	6.0414
7.8261	7.0708 + 3.3545i	-2.8455 + 0.8510i	2.9700
12.8901	-12.0861 - 4.4812i	-2.4756 - 1.7973i	3.0592

- We keep also the sum of squares of the remaining powers



	X	Q	||Q||
0.0000	0.0000	0.0000	0
12.8901	-12.0861 + 4.4812i	-2.4756 + 1.7973i	3.0592
7.8261	7.0708 - 3.3545i	-2.8455 - 0.8510i	2.9700
4.3321	0.8045 + 4.2567i	-5.5245 - 2.4452i	6.0414
2.0729	0.2386 + 2.0592i	-2.4342 - 1.9897i	3.1439
6.9035	-1.6003 + 6.7154i	-2.1082 + 1.6303i	2.6650
2.7897	-2.6539 + 0.8595i	-3.6438 - 1.9424i	4.1292
6.9880	5.8845 + 3.7689i	-1.5989 - 0.1514i	1.6061
4.4229	-2.0182 + 3.9356i	1.6186 - 2.3544i	2.8571
3.9548	-3.9484 + 0.2234i	15.2057 - 6.0515i	16.3656
2.7512	-1.2440 + 2.4538i	-3.6746 + 0.3413i	3.6904
1.9304	-1.6268 + 1.0393i	-0.0532 + 0.6724i	0.6745
4.2094	-1.0458 + 4.0774i	-1.8331 - 0.6000i	1.9288
4.2856	-4.2436 + 0.5988i	0.1642 + 1.3907i	1.4004
6.4725	-6.4020 - 0.9529i	-7.3632 - 5.7938i	9.3694
3.5360	-3.3663 - 1.0825i	-2.2362 - 1.7287i	2.8265
2.9264	-2.9264	-5.1339	5.1339
3.5360	-3.3663 + 1.0825i	-2.2362 + 1.7287i	2.8265
6.4725	-6.4020 + 0.9529i	-7.3632 + 5.7938i	9.3694
4.2856	-4.2436 - 0.5988i	0.1642 - 1.3907i	1.4004
4.2094	-1.0458 - 4.0774i	-1.8331 + 0.6000i	1.9288
1.9304	-1.6268 - 1.0393i	-0.0532 - 0.6724i	0.6745
2.7512	-1.2440 - 2.4538i	-3.6746 - 0.3413i	3.6904
3.9548	-3.9484 - 0.2234i	15.2057 + 6.0515i	16.3656
4.4229	-2.0182 - 3.9356i	1.6186 + 2.3544i	2.8571
6.9880	5.8845 - 3.7689i	-1.5989 + 0.1514i	1.6061
2.7897	-2.6539 - 0.8595i	-3.6438 + 1.9424i	4.1292
6.9035	-1.6003 - 6.7154i	-2.1082 - 1.6303i	2.6650
2.0729	0.2386 - 2.0592i	-2.4342 + 1.9897i	3.1439
4.3321	0.8045 - 4.2567i	-5.5245 + 2.4452i	6.0414
7.8261	7.0708 + 3.3545i	-2.8455 + 0.8510i	2.9700
12.8901	-12.0861 - 4.4812i	-2.4756 - 1.7973i	3.0592

- Calculate distance from k known coeffs



Optimize distance from remaining coefficients

minPower
6.9035

0.0000	0.0000	0.0000
12.8901	-12.0861 + 4.4812i	-2.4756 + 1.7973i
7.8261	7.0708 - 3.3545i	-2.8455 - 0.8510i
4.3321	0.8045 + 4.2567i	-5.5245 - 2.4452i
2.0729	0.2386 + 2.0592i	-2.4342 - 1.9897i
6.9035	-1.6003 + 6.7154i	-2.1082 + 1.6303i
2.7897	-2.6539 + 0.8595i	-3.6438 - 1.9424i
6.9880	5.8845 + 3.7689i	-1.5989 - 0.1514i
4.4229	-2.0182 + 3.9356i	1.6186 - 2.3544i
3.9548	-3.9484 + 0.2234i	15.2057 - 6.0515i
2.7512	-1.2440 + 2.4538i	-3.6746 + 0.3413i
1.9304	-1.6268 + 1.0393i	-0.0532 + 0.6724i
4.2094	-1.0458 + 4.0774i	-1.8331 - 0.6000i
4.2856	-4.2436 + 0.5988i	0.1642 + 1.3907i
6.4725	-6.4020 - 0.9529i	-7.3632 - 5.7938i
3.5360	-3.3663 - 1.0825i	-2.2362 - 1.7287i
2.9264	-2.9264	-5.1339
3.5360	-3.3663 + 1.0825i	-2.2362 + 1.7287i
6.4725	-6.4020 + 0.9529i	-7.3632 + 5.7938i
4.2856	-4.2436 - 0.5988i	0.1642 - 1.3907i
4.2094	-1.0458 - 4.0774i	-1.8331 + 0.6000i
1.9304	-1.6268 - 1.0393i	-0.0532 - 0.6724i
2.7512	-1.2440 - 2.4538i	-3.6746 - 0.3413i
3.9548	-3.9484 - 0.2234i	15.2057 + 6.0515i
4.4229	-2.0182 - 3.9356i	1.6186 + 2.3544i
6.9880	5.8845 - 3.7689i	-1.5989 + 0.1514i
2.7897	-2.6539 - 0.8595i	-3.6438 + 1.9424i
6.9035	-1.6003 - 6.7154i	-2.1082 - 1.6303i
2.0729	0.2386 - 2.0592i	-2.4342 + 1.9897i
4.3321	0.8045 - 4.2567i	-5.5245 + 2.4452i
7.8261	7.0708 + 3.3545i	-2.8455 + 0.8510i
12.8901	-12.0861 - 4.4812i	-2.4756 - 1.7973i

Optimization

$$\max \|\vec{X} - \vec{Q}\|^2 \text{ such that}$$

$$\sum_{k=1}^{m} \|X_k\|^2 = e^2$$

$$\|X_k\|^2 \leq minPower,\ k = 1, \ldots, m$$

Generalization

$$\min(\max) \quad \|\mathbf{X} - \mathbf{Q}\|_2$$

$$\text{s.t.} \quad |X_l| \leq \min_{j \in p_x^+} |X_j|, \quad \forall l \in p_x^-$$

$$|Q_l| \leq \min_{j \in p_q^+} |Q_j|, \quad \forall l \in p_q^-$$

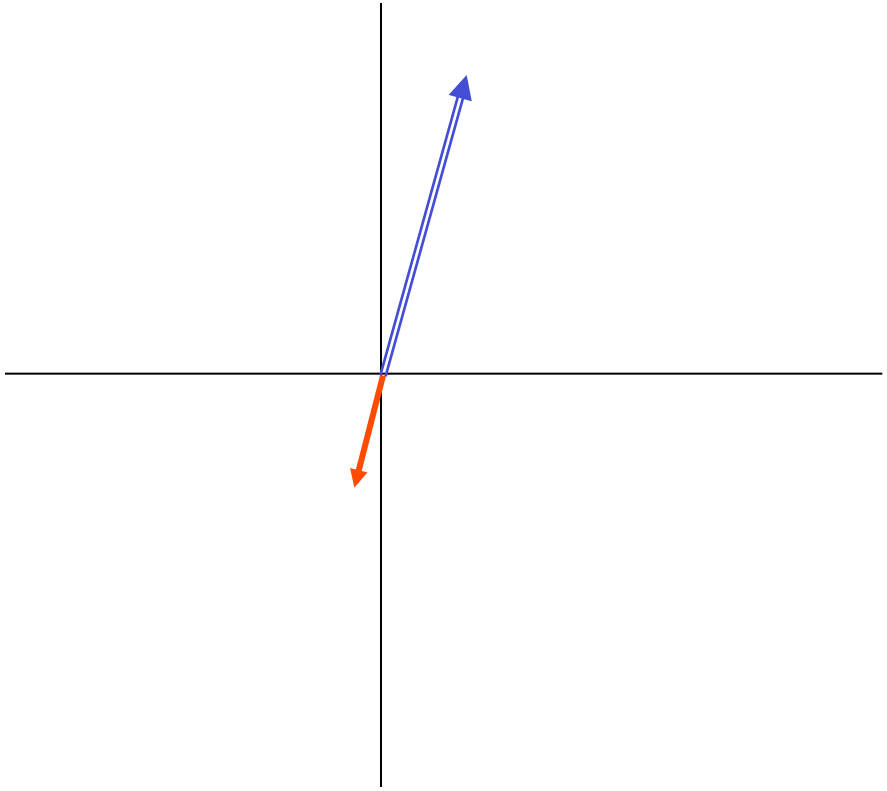
$$\sum_{l \in p_x^-} |X_l|^2 = e_x$$

$$\sum_{l \in p_q^-} |Q_l|^2 = e_q$$

$$\mathbf{X}^- \in \mathbb{C}^{|p_x^-|}, \mathbf{Q}^- \in \mathbb{C}^{|p_q^-|}$$

NEW YORK UNIVERSITY ABU DHABI

water-filling

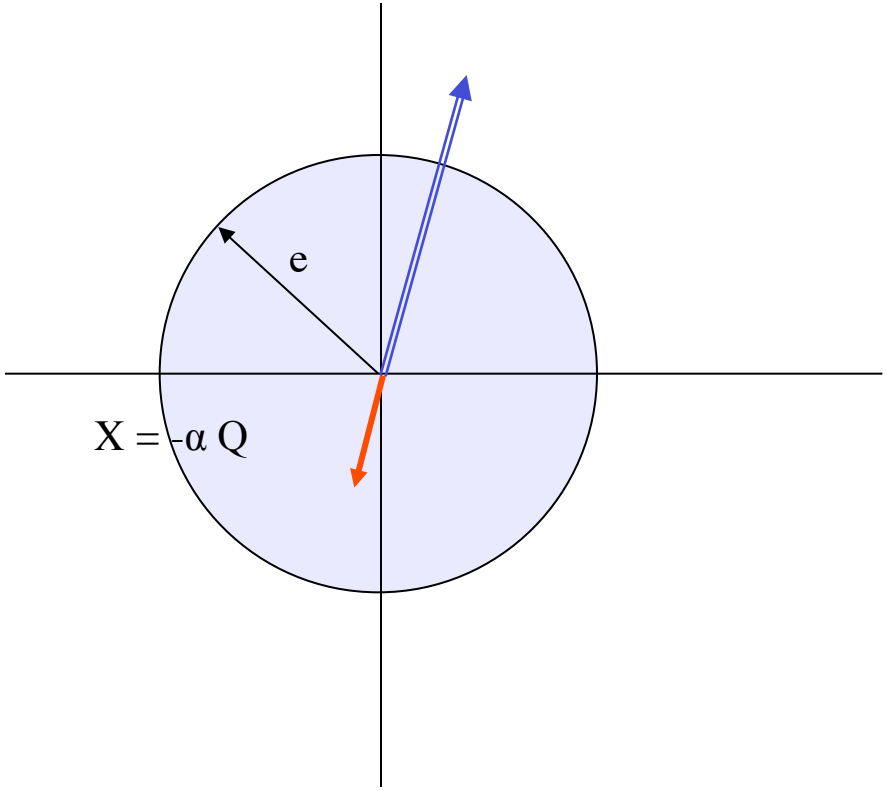


$$\max_{\mathbf{x}} \sum_i q_i x_i$$

$$\text{s.t.} \quad 0 \le x_i \le U_x$$

$$\sum_i x_i^2 = e_x$$

water-filling

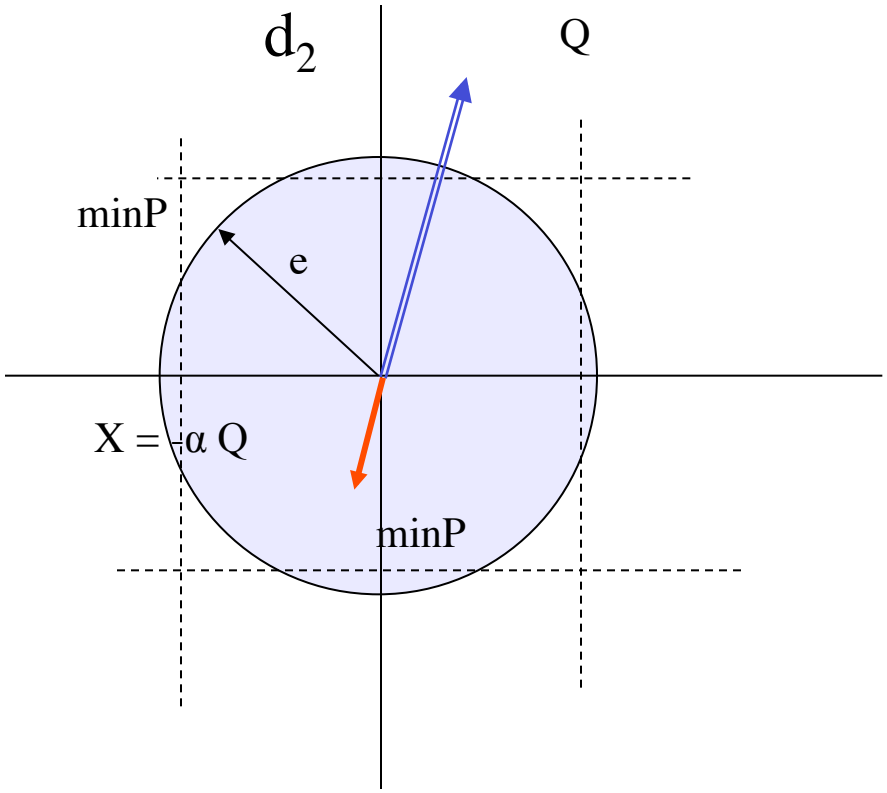


$$\max_{\mathbf{x}} \sum_i q_i x_i$$

$$\text{s.t.} \quad 0 \le x_i \le U_x$$

$$\sum_i x_i^2 = e_x$$

water-filling

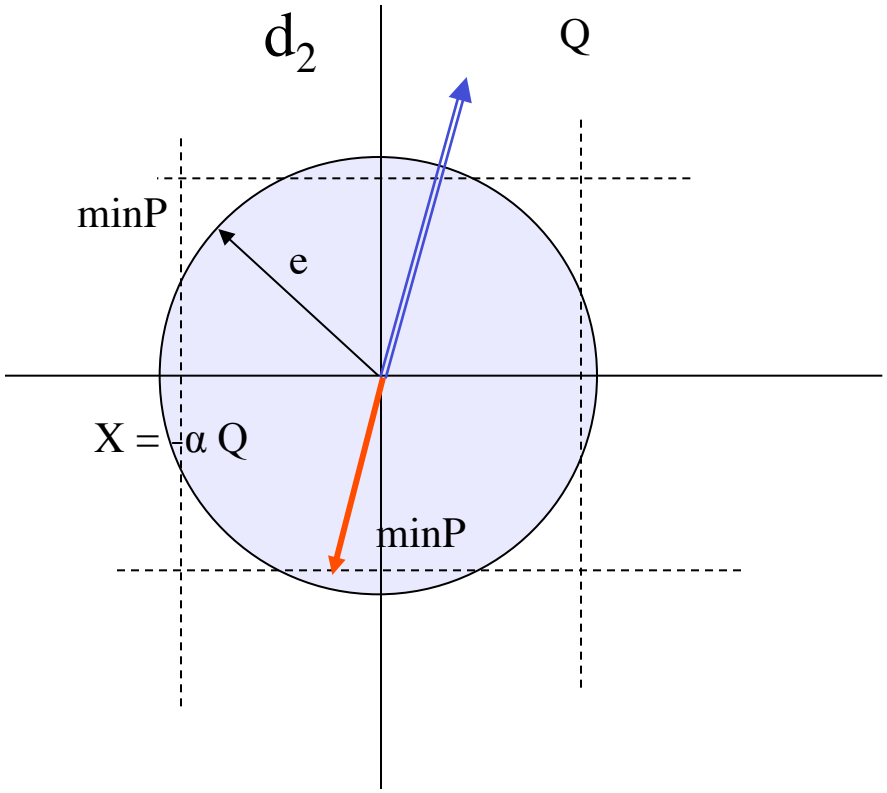


$$\max_{\mathbf{x}} \sum_i q_i x_i$$

$$\text{s.t.} \quad 0 \le x_i \le U_x$$

$$\sum_i x_i^2 = e_x$$

water-filling

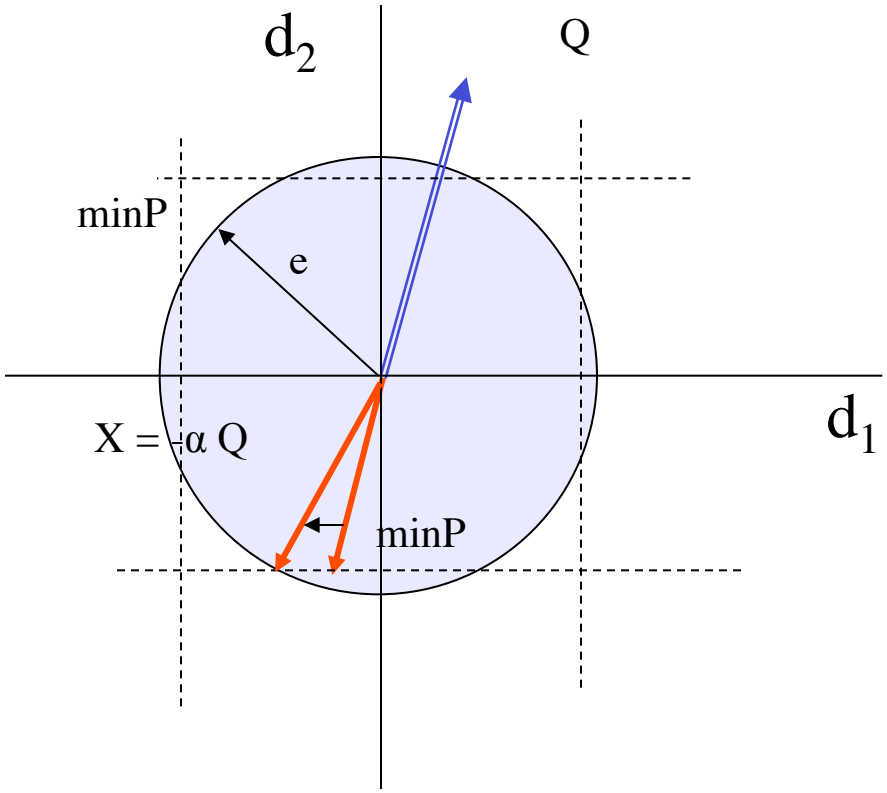


$$\max_{\mathbf{x}} \sum_i q_i x_i$$

$$\text{s.t.} \quad 0 \le x_i \le U_x$$

$$\sum_i x_i^2 = e_x$$

water-filling

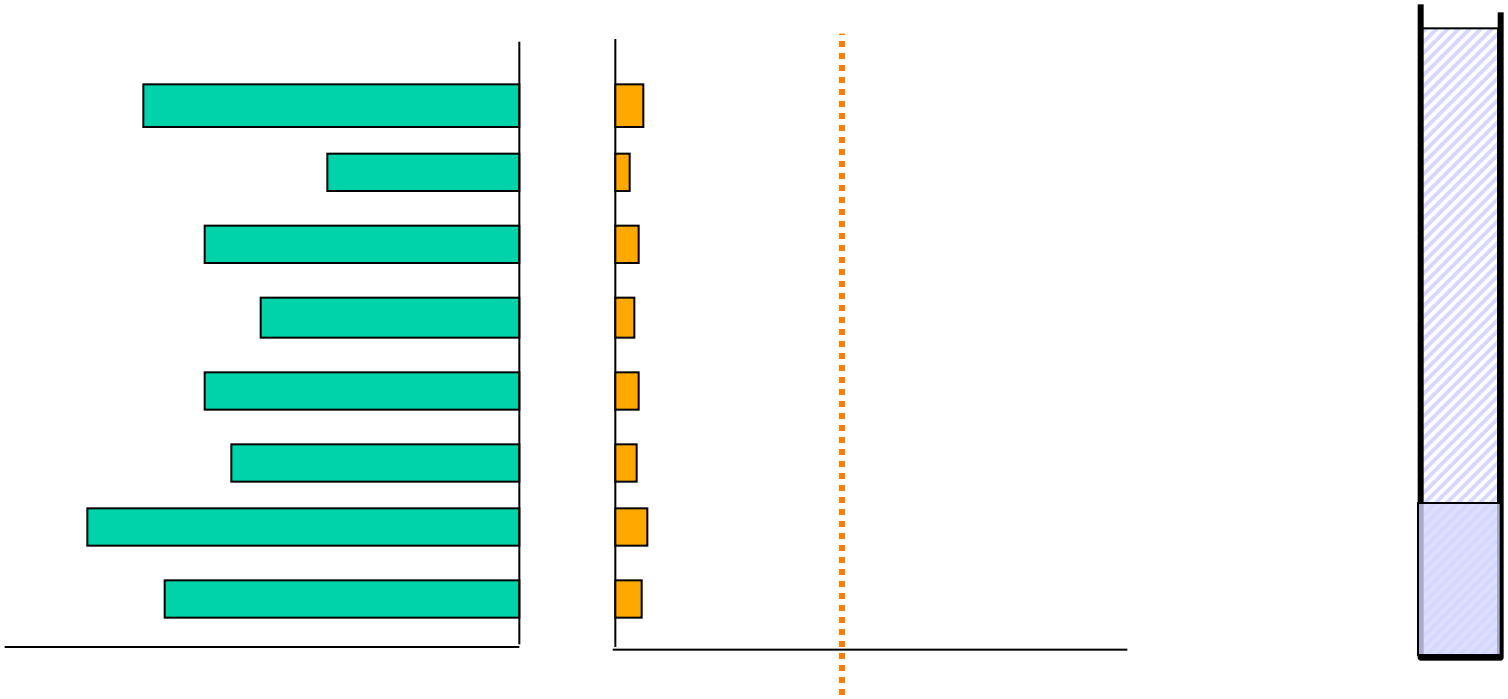


$$\max_{\mathbf{x}} \sum_i q_i x_i$$

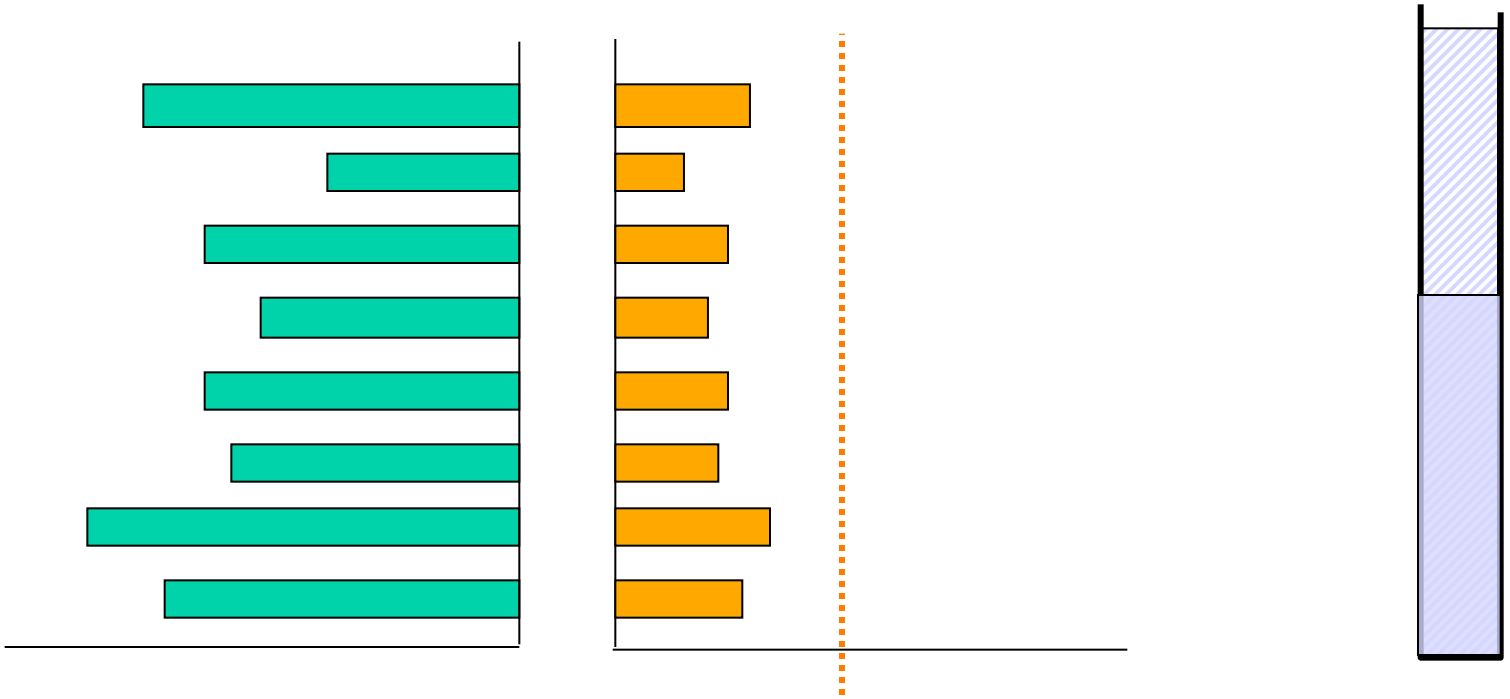
$$\text{s.t.} \quad 0 \le x_i \le U_x$$

$$\sum_i x_i^2 = e_x$$

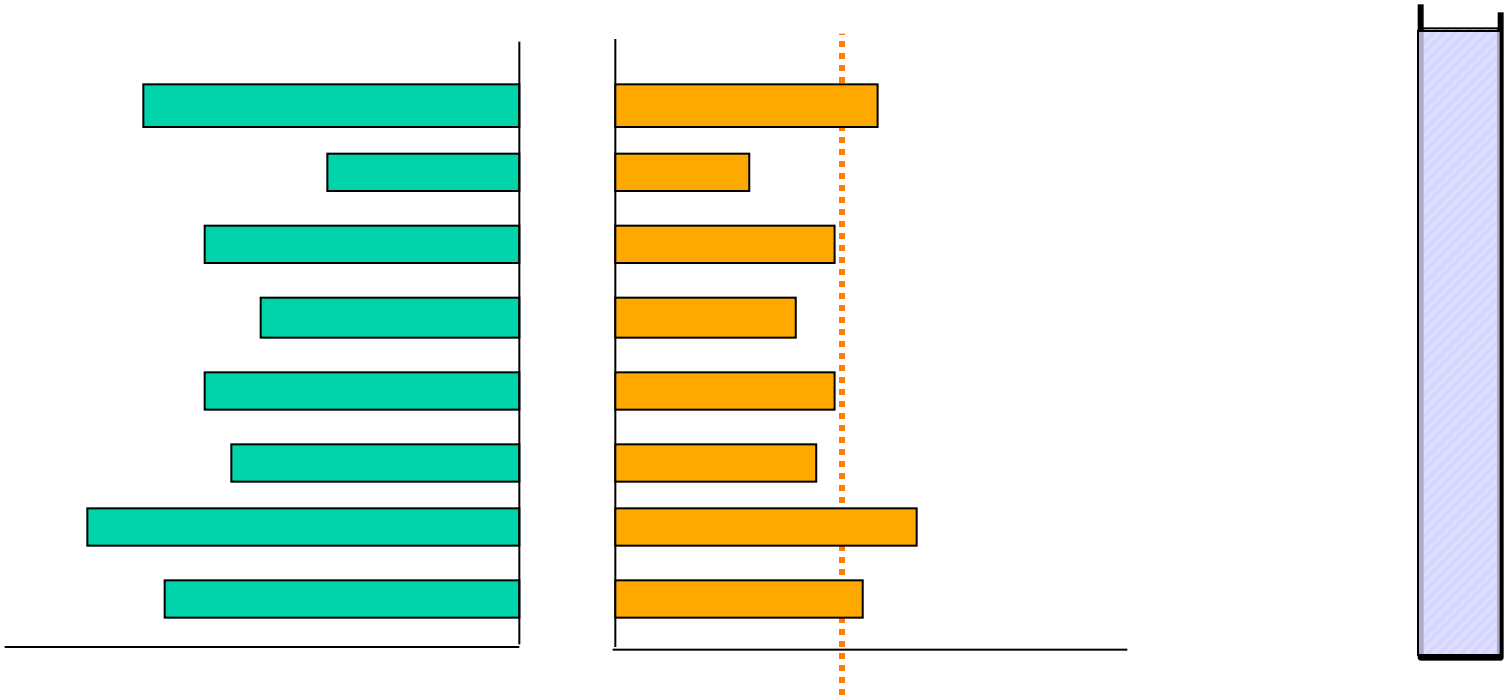
in N-dimensions



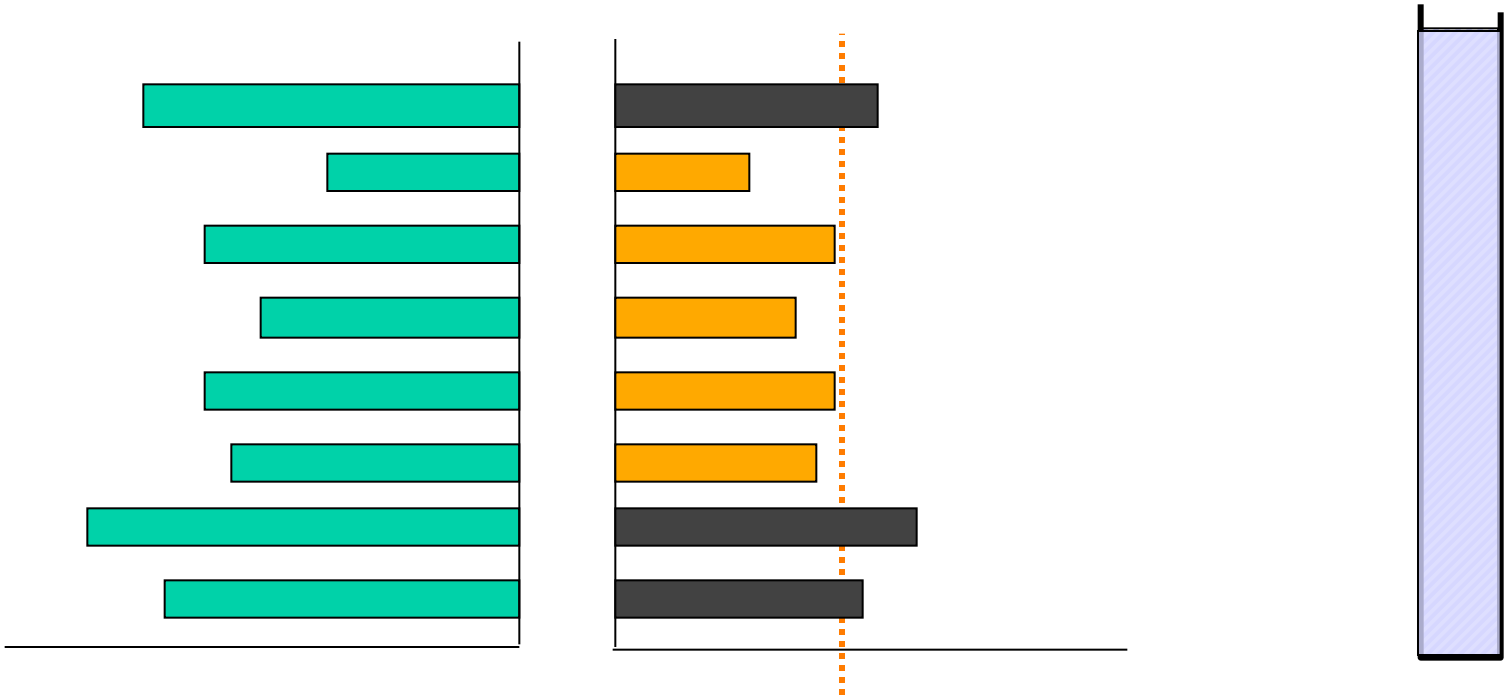
in N-dimensions



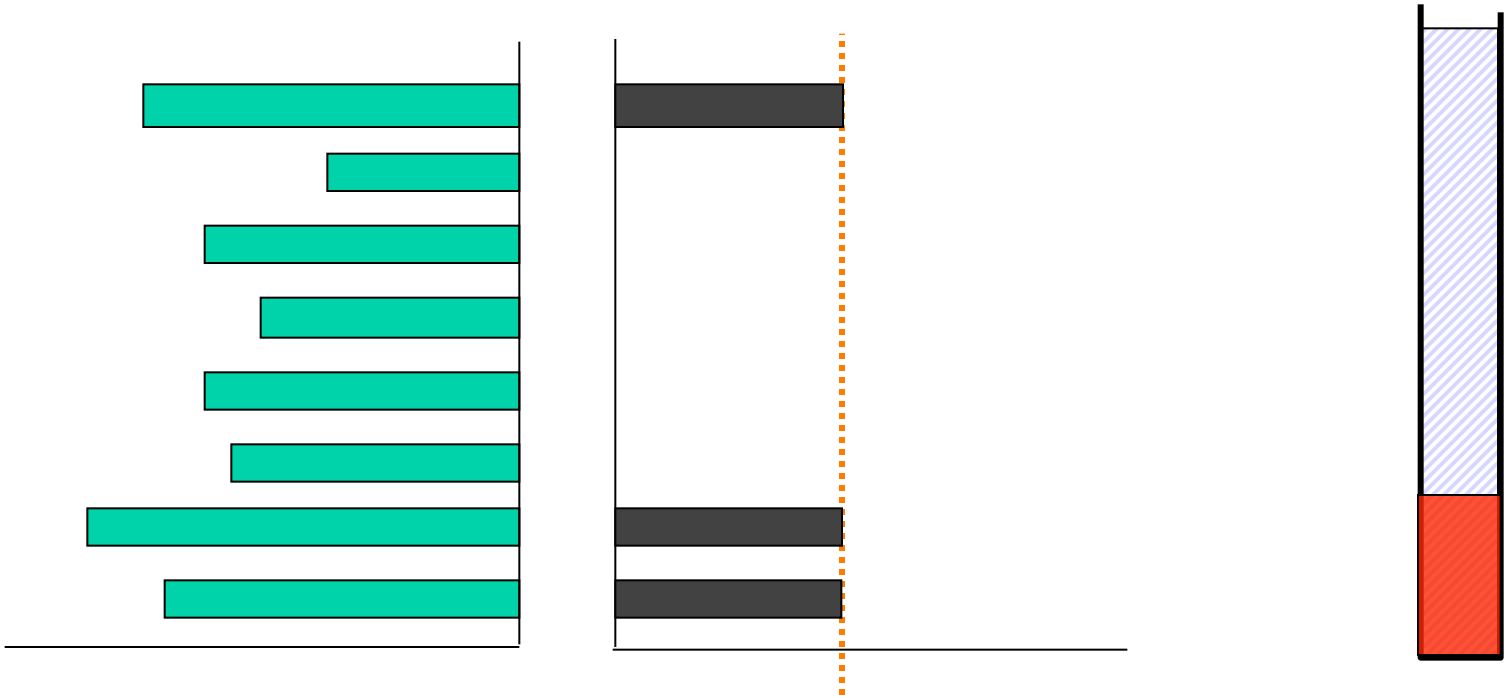
in N-dimensions



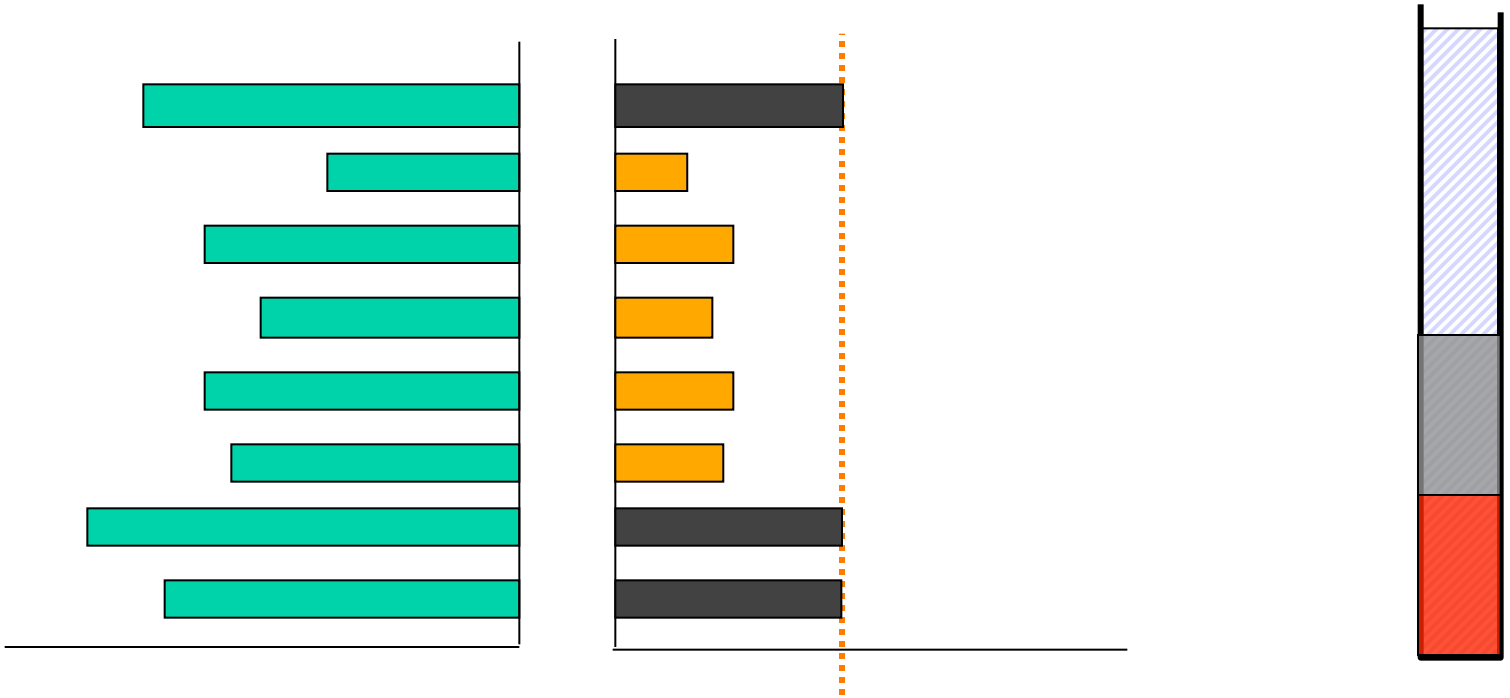
in N-dimensions



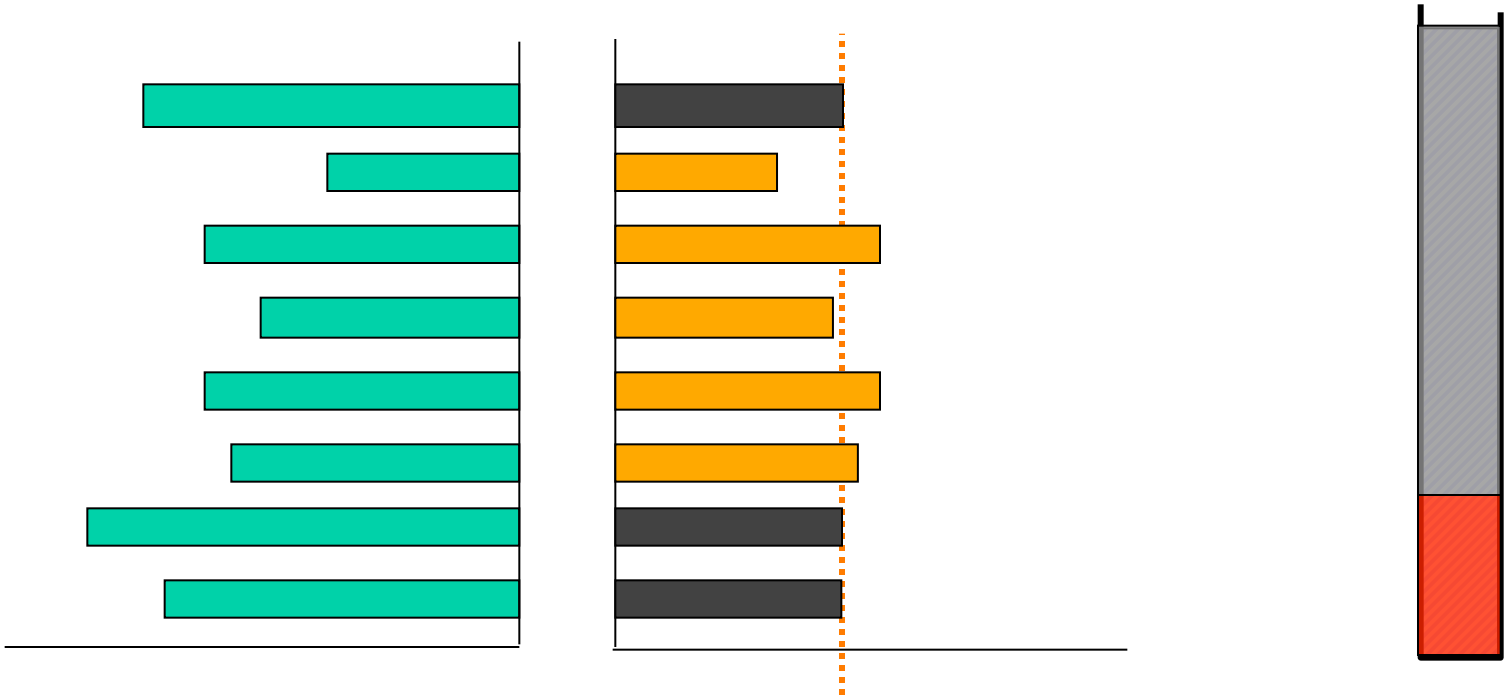
in N-dimensions



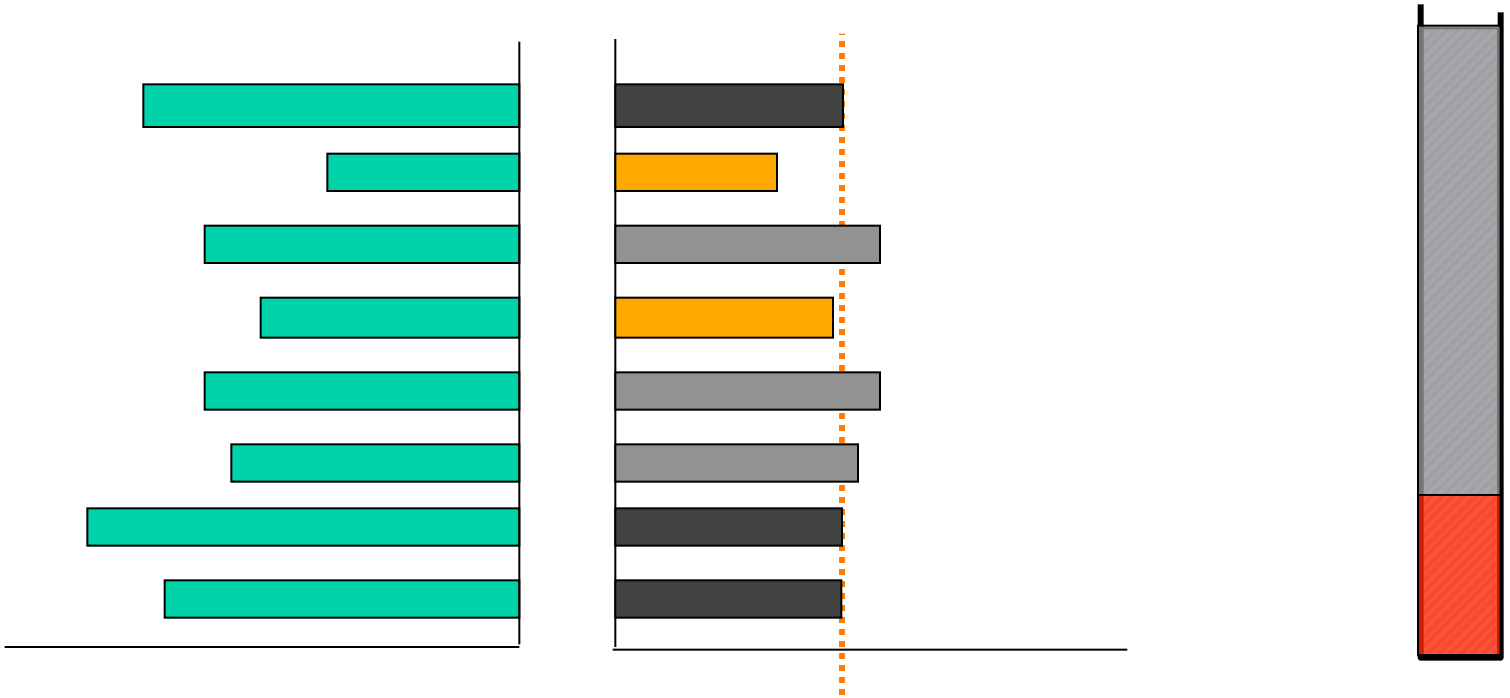
in N-dimensions



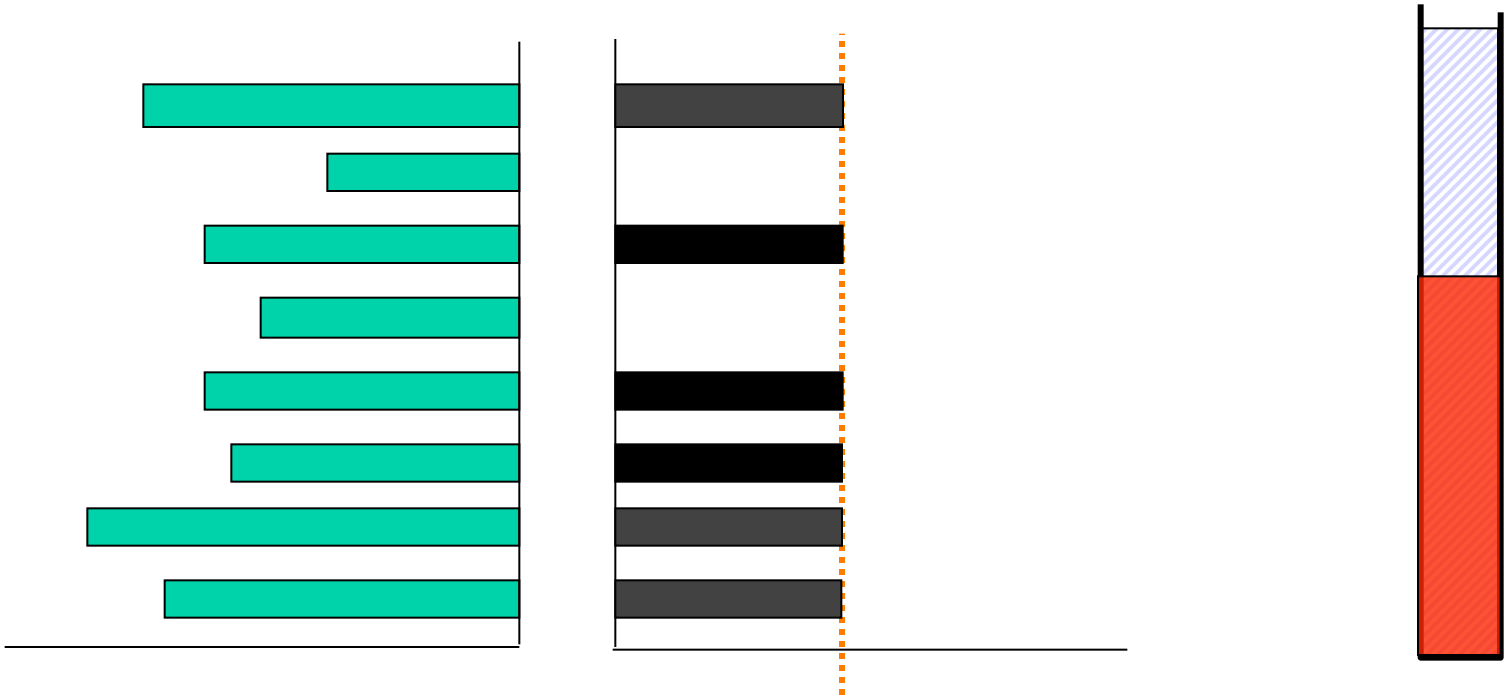
in N-dimensions



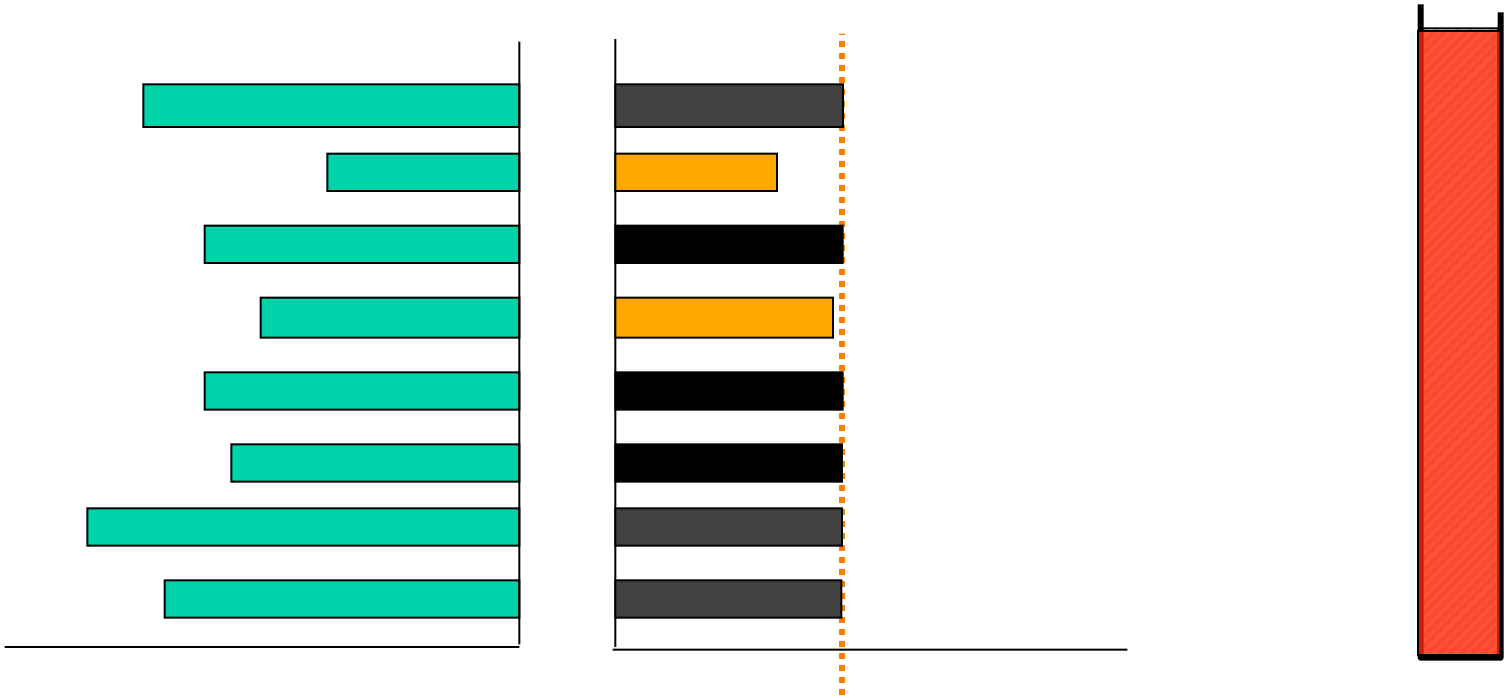
in N-dimensions



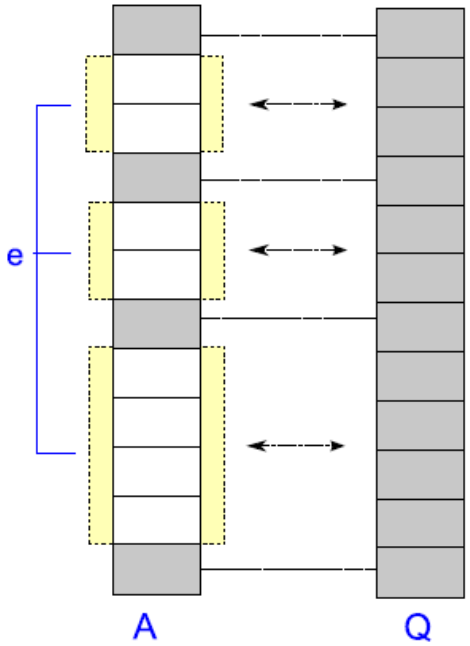
in N-dimensional space



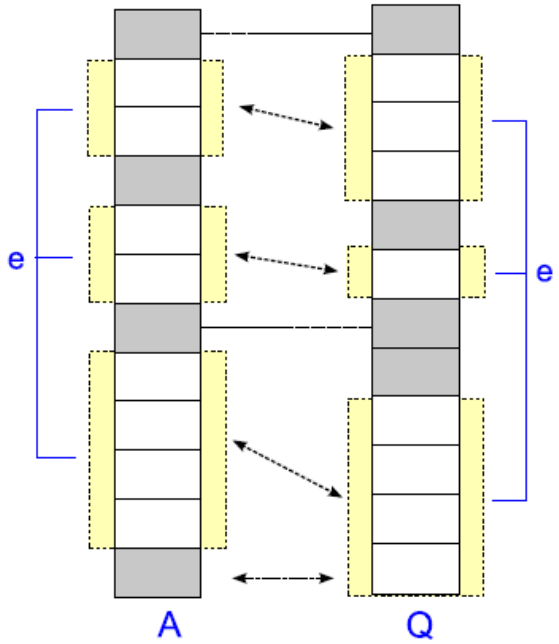
in N-dimensional space



When both sequences are compressed the optimal distance estimation can be solved using a *double waterfilling* process.



single waterfilling



double waterfilling

the bounds are optimally tight

Theorem (Freris et al. ’12): The computation of lower and upper bounds given the aforementioned compressed representations can be solved exactly using double water-filling problem. The lower and upper bounds are optimally tight; no tighter solutions can be provided

Key concept

$$\max_{\mathbf{x},\mathbf{q}} \sum_{i\in p_q^+\cap p_x^+} q_i x_i + \sum_{i\in p_x^-\cap p_q^+} q_i x_i + \sum_{i\in p_q^-\cap p_x^+} q_i x_i + \sum_{i\in p_q^-\cap p_x^-} q_i x_i$$

s.t. $x_i \in [0, U_x], q_i \in [0, U_q]$

$$\sum_i x_i^2 = e_x, \quad \sum_i q_i^2 = e_q$$

Energy allocation

$$\max_{\mathbf{x},\mathbf{q}} \underbrace{\sum_{i\in p_x^-\cap p_q^+} q_i x_i}_{\substack{x_i\in[0,U_x] \\ \sum_i x_i^2 = e_x^{(1)}}} + \underbrace{\sum_{i\in p_q^-\cap p_x^+} q_i x_i}_{\substack{q_i\in[0,U_q] \\ \sum_i q_i^2 = e_q^{(1)}}} + \underbrace{\sum_{i\in p_q^-\cap p_x^-} q_i x_i}_{\substack{\sum_i x_i^2 = e_x - e_x^{(1)} \\ \sum_i q_i^2 = e_q - e_q^{(1)}}}$$

decoupled waterfillings

C-S

$$\sqrt{\mathbf{e_x} - \mathbf{e_x^{(1)}}}\sqrt{\mathbf{e_q} - \mathbf{e_q^{(1)}}}$$

Optimization over two positive values $e_x^{(1)}, e_q^{(1)}$

..we can find the exact solution

NEW YORK UNIVERSITY ABU DHABI

Double Water-filling algorithm

Double water-filling algorithm
Inputs: $\{b_i\}_{i\in P_1}, \{a_i\}_{i\in P_2}, e_x, e_q, A, B$
Outputs: $\{a_i, \alpha_i\}_{i\in p_x^-}, \{b_i, \beta_i\}_{i\in p_q^-}, \lambda, \mu, v_{\text{opt}}$

1. if $p_x^- \cap p_q^- = \emptyset$ then use water-filling algorithm; return; endif
2. if $p_x^- = p_q^-$ then set $a_l = \sqrt{\frac{e_x}{|P_3|}}, b_l = \sqrt{\frac{e_q}{|P_3|}}$, $\alpha_l = \beta_l = 0$ for all $l \in p_x^-$, $v_{\text{opt}} = -\sqrt{e_x}\sqrt{e_q}$; return; endif
3. if $e_x \le |P_1|A^2$ and $e_q \le |P_2|B^2$ then

$$\{a_l\}_{l\in P_1} = \text{waterfill } (\{b_l\}_{l\in P_1}, e_x, A)$$
$$\{b_l\}_{l\in P_2} = \text{waterfill } (\{a_l\}_{l\in P_2}, e_q, B)$$

with optimal values $v_{\text{opt}}^{(a)}, v_{\text{opt}}^{(b)}$ respectively

4. Set $a_l = b_l = \alpha_l = \beta_l = 0$ for all $l \in P_3$, $v_{\text{opt}} = -v_{\text{opt}}^{(a)} - v_{\text{opt}}^{(b)}$; return;
5. endif
6. Calculate the unique root $\bar{\gamma}$ of

$$\frac{e_x - \sum_{l\in P_1} \min(b_l^2\gamma, A^2)}{e_q - \sum_{l\in P_2} \min(a_l^2 \frac{1}{\gamma}, B^2)} = \gamma$$

and define

$$e_x' = e_x - \sum_{l\in P_1} \min(b_l^2\bar{\gamma}, A^2)$$
$$e_q' = e_q - \sum_{l\in P_2} \min(a_l^2 \frac{1}{\bar{\gamma}}, B^2)$$

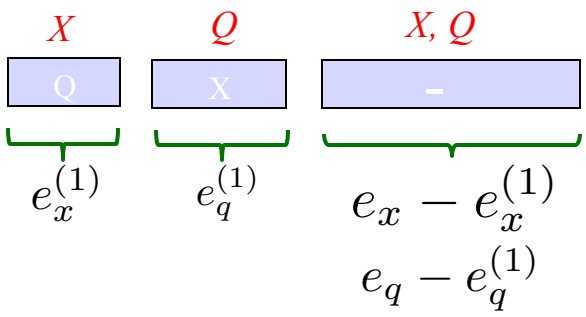
7. Set

$$\{a_l\}_{l\in P_1} = \text{waterfill } (\{b_l\}_{l\in P_1}, e_x - e_x', A)$$
$$\{b_l\}_{l\in P_2} = \text{waterfill } (\{a_l\}_{l\in P_2}, e_q - e_q', B)$$

with optimal values $v_{opt}^{(a)}, v_{opt}^{(b)}$ respectively

8. Set $a_l = \sqrt{\frac{e_x'}{|P_3|}}, b_l = \sqrt{\frac{e_q'}{|P_3|}}, \alpha_l = \beta_l = 0, l \in P_3$ and set $v_{\text{opt}} = -v_{opt}^{(a)} - v_{opt}^{(b)} - \sqrt{e_x'}\sqrt{e_q'}$

Linear equations



Intuition:

Water-fill for the discarded coefficients of the two vectors separately..

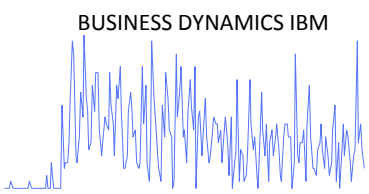
..using the *appropriate* energy allocation

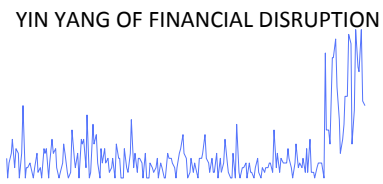
Zero-overhead algorithm
Complexity: Θ(nlogn)

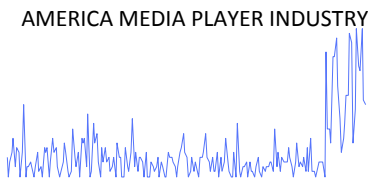
Experiments

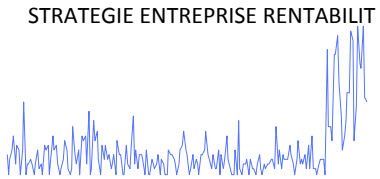
Unica database. IBM web traffic for year of 2010

- Marketing/Adwords recommendation
  - Analysis/Storage of weblog queries (1TB of data per month)
  - GBS: Scheduling advertising campaigns / pricing

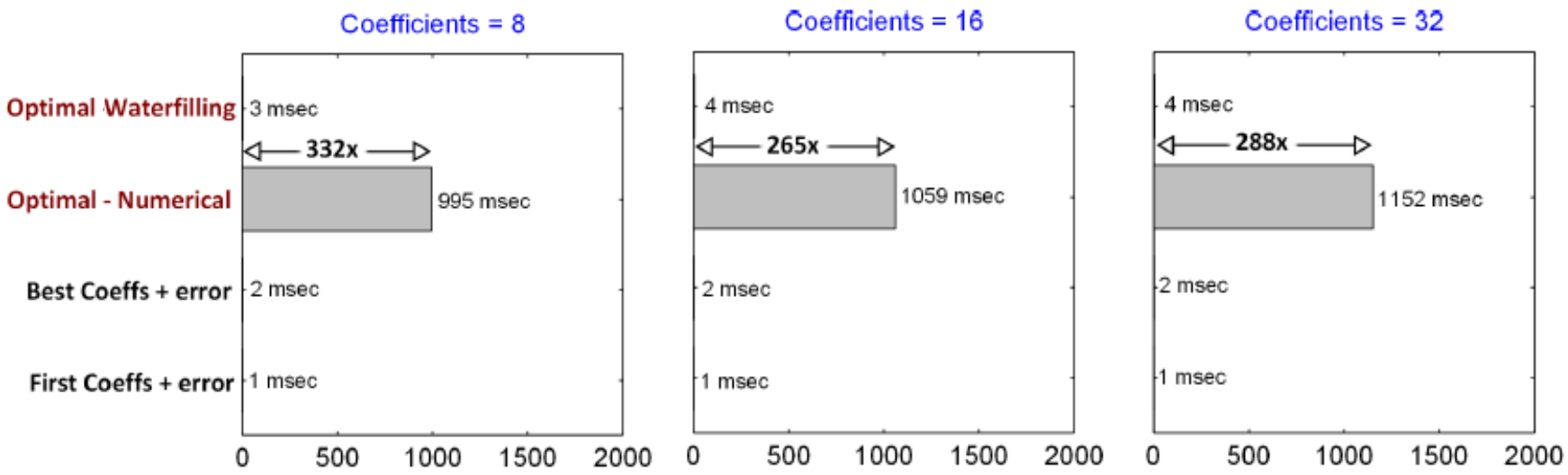




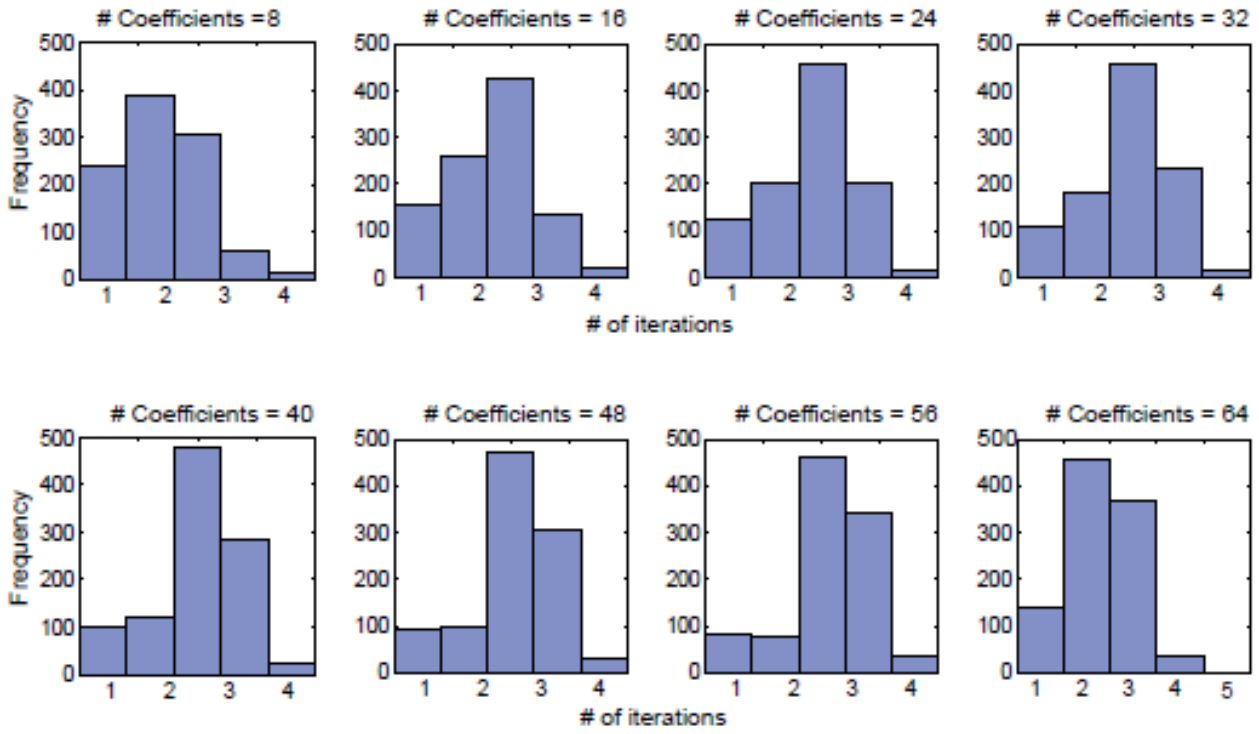




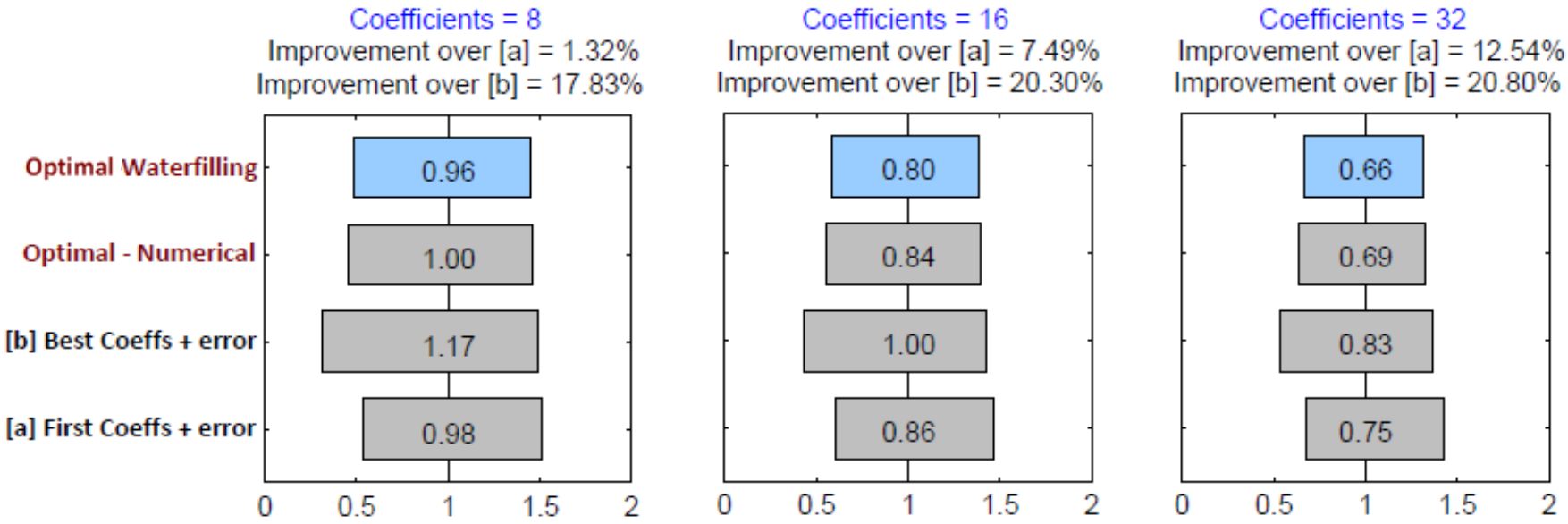
our analytic solution is 300x faster than convex optimizers



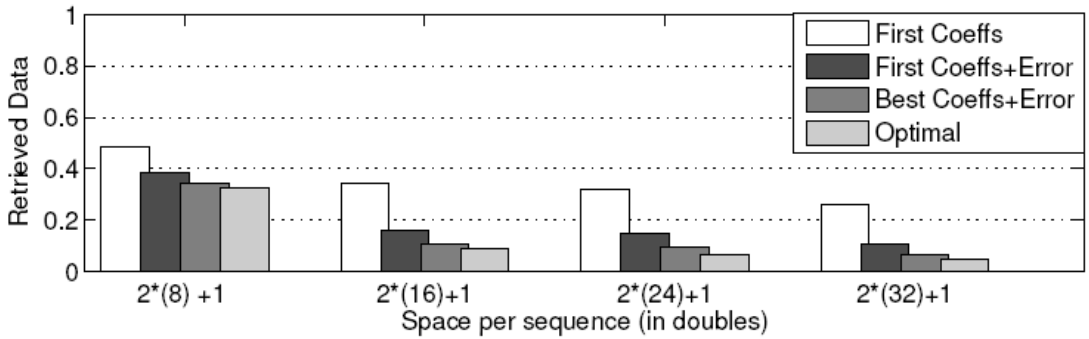
Our algorithm converges very fast: in 2-3 iterations irrespective of the compression rate



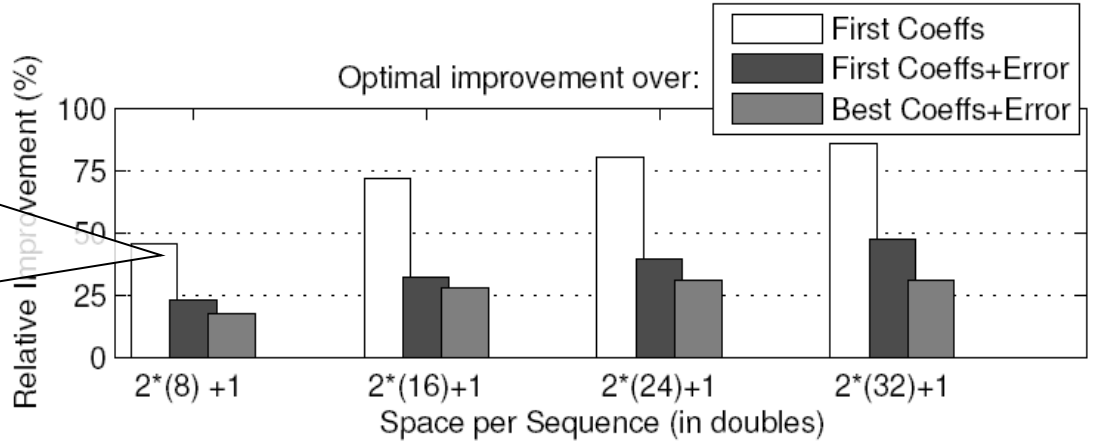
the derived LB/UB are 20% tighter than state-of-art



when searching in a DB this results in up to 80% data pruning



We retrieve 20%-80% fewer sequences than other approaches



NEW YORK UNIVERSITY ABU DHABI

Conclusions

- Increasing data sizes is a perennial problem for data analysis
  - It is important to support mining directly on the compressed data
- We have shown an exact algorithm for obtaining optimally tight Euclidean bounds on compressed representations

Mining directly on compressed data with provable guarantees

Generalizations / Applications

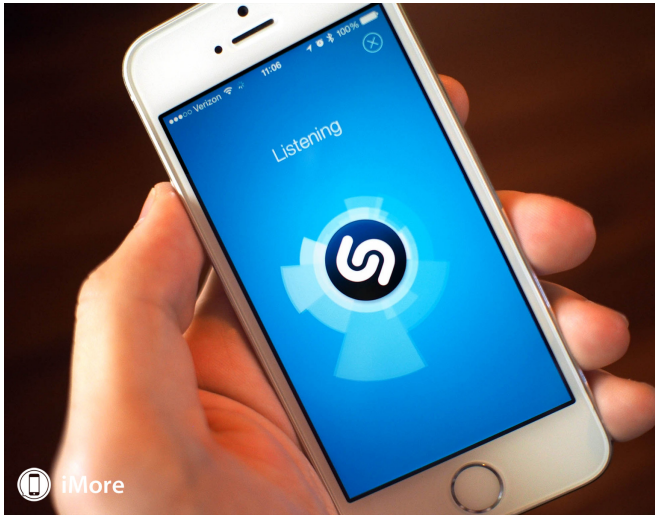
Cosine Similarity (text documents):

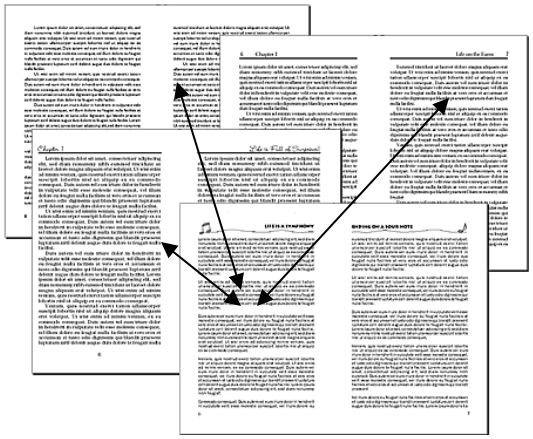
$\cos(x,y) = 1 - L_2(x,y)^2/2$

Correlation (financial analysis):

$\text{corr}(x,y) = 1 - L_2(x,y)^2/2$ (for normalized signals x,y)

Dynamic Time Warping (flexible similarity metric)





References

- M. Vlachos, N. Freris and A. Kyrillidis, **"Compressive Mining: Fast and Optimal Data Mining in the Compressed Domain."** International Journal on Very Large Data Bases (VLDBJ), 2014.

- N. Freris, M. Vlachos and F. Fusco **"Method for distance estimation on compressed data."** US Patent WO2013160840 A1, 2013.

- N. Freris, M. Vlachos and S. Kozat, **"Optimal Distance Estimation Between Compressed Data Series."** Proceedings of the SIAM International Conference on Data Mining (SDM12), pp. 343-354, 2012.

Thank you

Questions…