Orthogonal Matching Pursuit
A Machine Learning Perspective

Slides: Team RAAAK

Outliers

Outliers in Linear Models

Samples that deviate strongly from the general pattern of our data

Outliers in Linear Models

Samples that deviate strongly from the general pattern of our data

Not well explained by our linear model

Outliers in Linear Models

- Samples that deviate strongly from the general pattern of our data
- Not well explained by our linear model
- Sources of outliers include:
  - Errors in measurement or data entry

Outliers in Linear Models

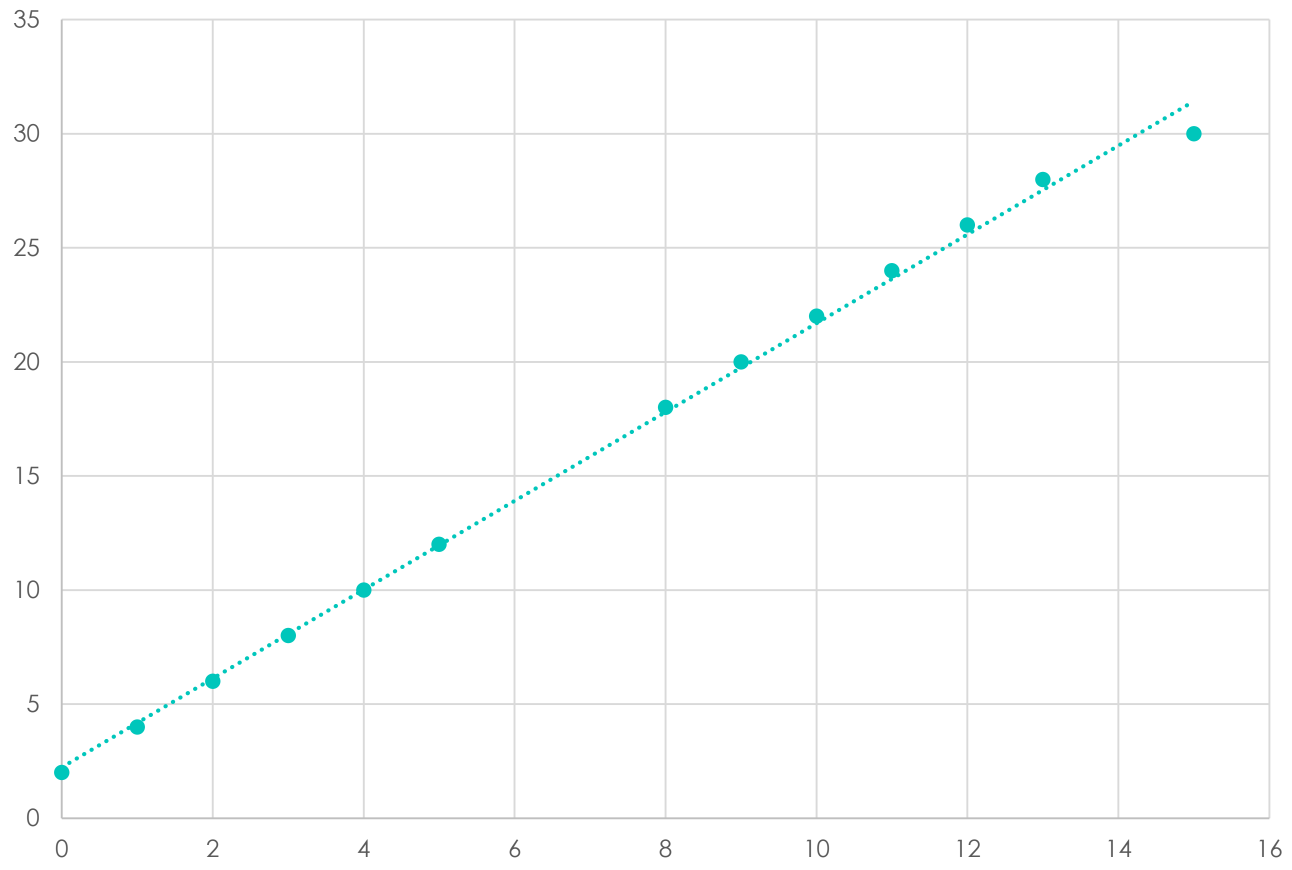
- Samples that deviate strongly from the general pattern of our data
- Not well explained by our linear model
- Sources of outliers include:
  - Errors in measurement or data entry
  - Sampling errors

Outliers in Linear Models

- Samples that deviate strongly from the general pattern of our data
- Not well explained by our linear model
- Sources of outliers include:
  - Errors in measurement or data entry
  - Sampling errors
  - Noise in our data that corrupts our true observation

The Impact of Outliers

- Outliers can drive up the error of our model!
- Cause us to not represent the true pattern of our data

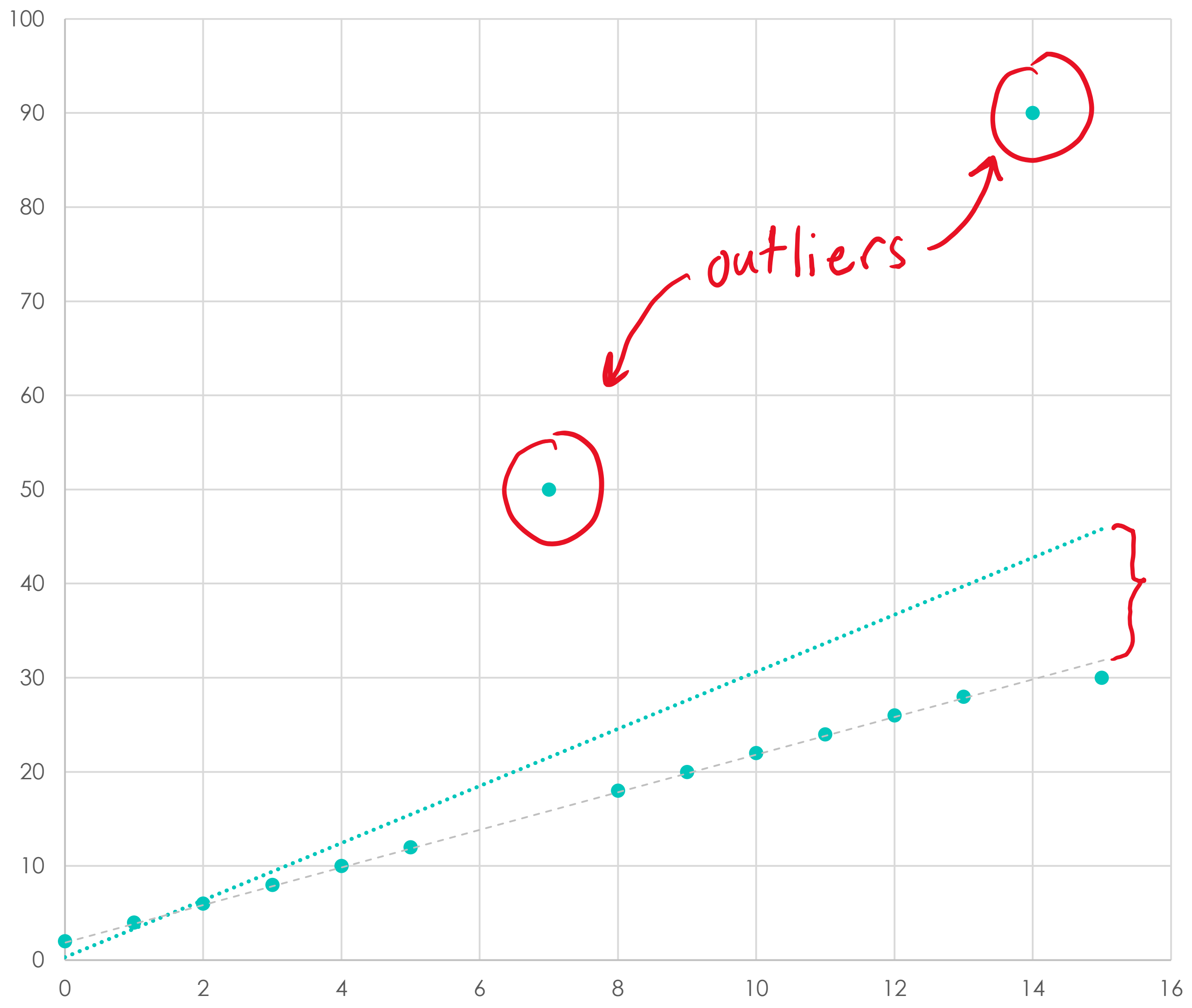


The Impact of Outliers

- Outliers can drive up the error of our model!
- Cause us to not represent the true pattern of our data

The Impact of Outliers

- Outliers can drive up the error of our model!
- Cause us to not represent the true pattern of our data



Facts About OMP – A 16A Review

01

Goal: Solve a LS problem to get a sparse solution

02

Sparse: containing mostly zero entries

03

K-sparse vector: k controls the number of nonzero entries

04

Assumption: the columns of X (data matrix) are normalized

05

Property: OMP is a Greedy Algorithm

OMP Review – Signal Processing Context

- You have seen this interpretation in EECS16A
- Context:
  - m satellites that are potentially broadcasting unique signals
  - A received signal that is a linear combination of a subset of these signals

OMP Review – Signal Processing Context

- You have seen this interpretation in EECS16A
- What we're given:
  - m satellites that are potentially broadcasting unique signals
  - A received signal that is a linear combination of a subset of these signals
- Goal
  - find which k satellites are transmitting signals that are present in our received signal

$$\boldsymbol{y} = \sum_{i \in K} \alpha_i \boldsymbol{s_i}^{(\tau_i)}$$

OMP Review - Algorithm

Algorithm 1: OMP Algorithm from 16A

Input: Codes $\boldsymbol{s}_i$, Received signal $\boldsymbol{y}$, Sparsity k, Residual Threshold ϵ

Output: The indices of the used codes F, the sent messages $\boldsymbol{x}$

$\boldsymbol{e} = \boldsymbol{y}$, j = 1, A = [], F = ∅;

while $j \leq k$ *and* $||\mathbf{e}|| \geq \epsilon$ **do**

  i, t = FindMaxCrossCorrelation();

  $F = F \bigcup \{i\}$;

  $A = [A|s_i^{(t)}]$;

  $\boldsymbol{x} = (A^T A)^{-1} A^T \boldsymbol{y}$;

  $\boldsymbol{e} = \boldsymbol{y} - A\boldsymbol{x}$;

  j = j + 1;

end

return $F, \boldsymbol{x}$

OMP Review - Algorithm

Algorithm 1: OMP Algorithm from 16A

Input: Codes $\boldsymbol{s}_i$, Received signal $\boldsymbol{y}$, Sparsity k, Residual Threshold ϵ

Output: The indices of the used codes F, the sent messages $\boldsymbol{x}$

$\boldsymbol{e} = \boldsymbol{y}$, j = 1, A = [], F = $\emptyset$;

while $j \leq k$ *and* $||\mathbf{e}|| \geq \epsilon$ **do**

  i, t = FindMaxCrossCorrelation();

  $F = F \bigcup \{i\}$;

  $A = [A|s_i^{(t)}]$;

  $\boldsymbol{x} = (A^T A)^{-1} A^T \boldsymbol{y}$;

  $\boldsymbol{e} = \boldsymbol{y} - A\boldsymbol{x}$;

  j = j + 1;

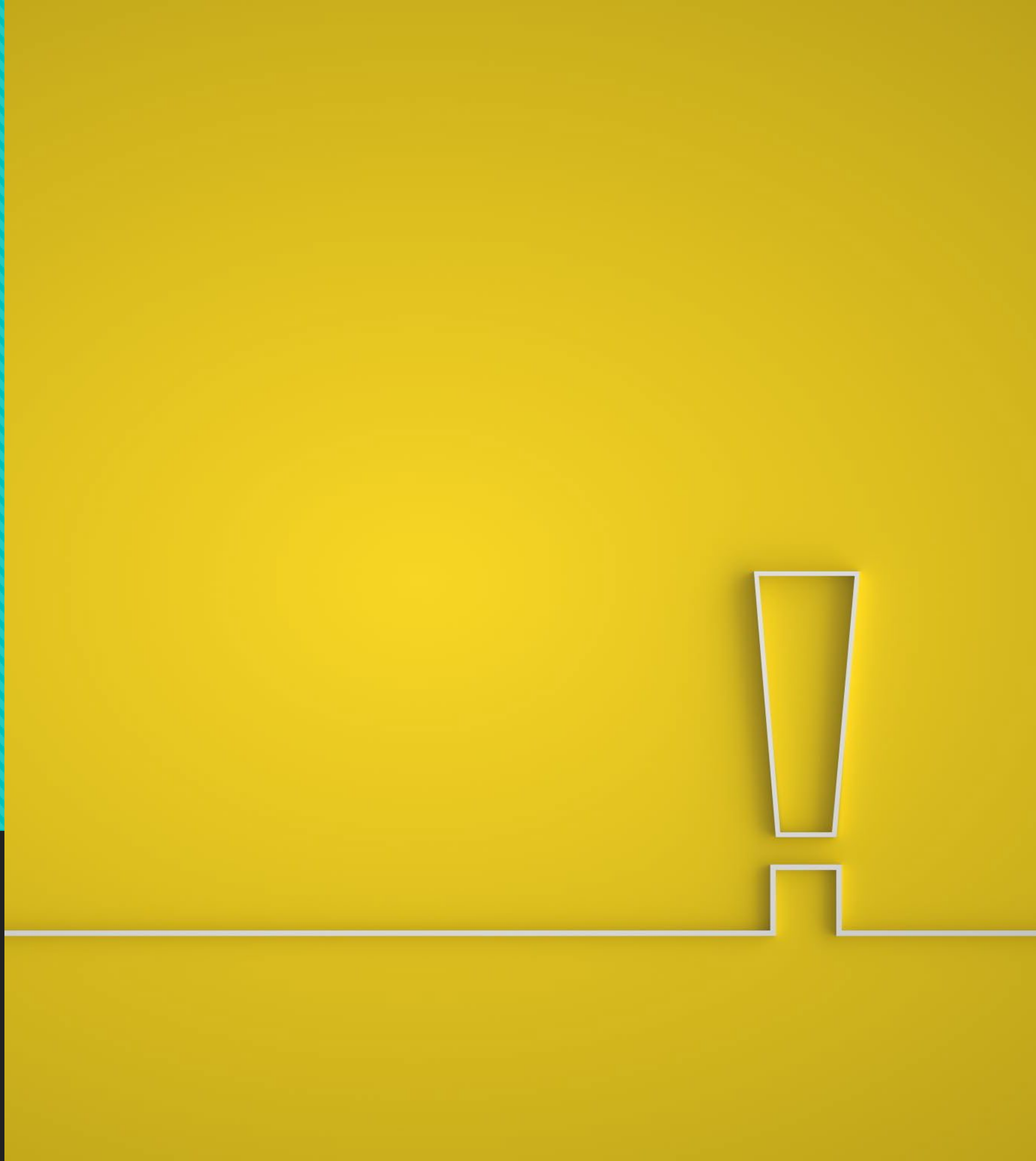
end

return $F, \boldsymbol{x}$

Recall: We find the code and time shift with the maximum cross-correlation to predict one satellite that contributes to our received signal

The Problem of Outlier Detection

We would like to **identify** and **remove** outlier data points to decrease model error



Outlier Detection: An Analogous Context

- Codes
  - In our new problem, codes correspond to our matrix $\boldsymbol{X}$ of data

Outlier Detection: An Analogous Context

- Codes
  - In our new problem, codes correspond to our matrix $\boldsymbol{X}$ of data
- Message α
  - Rather than recover a message broadcasted by satellites, we aim to learn an appropriate set of weights $\boldsymbol{w}$

Outlier Detection: An Analogous Context

- Codes
  - In our new problem, codes correspond to our matrix $\boldsymbol{X}$ of data
- Message α
  - Rather than recover a message broadcasted by satellites, we aim to learn an appropriate set of weights $\boldsymbol{w}$
- Time shifts τ_i
  - In Outlier Detection, we no longer have the problem of time delays that we saw in the signal processing context
  - This means we no longer have to compute any cross-correlation values

Outlier Detection: An Analogous Context

- Codes
  - In our new problem, codes correspond to our matrix $\boldsymbol{X}$ of data
- Message α
  - Rather than recover a message broadcasted by satellites, we aim to learn an appropriate set of weights $\boldsymbol{w}$
- Time shifts τ_i
  - In Outlier Detection, we no longer have the problem of time delays that we saw in the signal processing context
  - This means we no longer have to compute any cross-correlation values
- Meaning of k
  - Previously, we thought of k as related to the number of satellites transmitting unique codes
  - Now, k refers to the **number of outliers we aim to detect**

OMP for Outlier Detection – Initial Setup

targets

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x_1}^T \\ \boldsymbol{x_2}^T \\ \vdots \\ \boldsymbol{x_n}^T \end{bmatrix} \boldsymbol{w} + \epsilon.$$

OMP for Outlier Detection – Initial Setup

Recall: we are given data in $(\vec{x}_i, y_i)$ pairs

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x_1}^T \\ \boldsymbol{x_2}^T \\ \vdots \\ \boldsymbol{x_n}^T \end{bmatrix} \boldsymbol{w} + \epsilon.$$

data matrix X

OMP for Outlier Detection – Initial Setup

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x_1}^T \\ \boldsymbol{x_2}^T \\ \vdots \\ \boldsymbol{x_n}^T \end{bmatrix} \boldsymbol{w} + \epsilon.$$

weights we aim to learn through Least Squares

OMP for Outlier Detection – Initial Setup

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{x_1}^T \\ \boldsymbol{x_2}^T \\ \vdots \\ \boldsymbol{x_n}^T \end{bmatrix} \boldsymbol{w} + \epsilon.$$

$$\begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

noise!

OMP for Outlier Detection – Error & Residuals

- Consider an error term ϵ_i added to each predicted output

$$y_i = \boldsymbol{x}_i^T \boldsymbol{w} + \boldsymbol{\epsilon}_i$$

OMP for Outlier Detection – Error & Residuals

- Consider an error term ϵ_i added to each predicted output

$$y_i = \boldsymbol{x_i^T w} + \boldsymbol{\epsilon_i}$$

- Constructing the error vector $\boldsymbol{\epsilon}$

$$\boldsymbol{y} = \boldsymbol{Xw} + \sum \epsilon_i \boldsymbol{e_i}$$

OMP for Outlier Detection – Error & Residuals

- Consider an error term ϵ_i added to each predicted output

$$y_i = \boldsymbol{x}_i^T \boldsymbol{w} + \boldsymbol{\epsilon_i}$$

- Constructing the error vector $\boldsymbol{\epsilon}$

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w} + \sum \epsilon_i \boldsymbol{e_i}$$

- Matrix-vector product form

$$[\boldsymbol{X}\ \boldsymbol{I}]\begin{bmatrix}\boldsymbol{w} \\ \boldsymbol{\epsilon}\end{bmatrix}$$

Recall:

$$\vec{e}_i = \begin{array}{cl} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} & \begin{array}{l} \\ \\ \\ \leftarrow i\text{th index} \\ \\ \\ \\ \end{array} \end{array}$$

OMP for Outlier Detection – Error & Residuals

$$[\boldsymbol{X}\ \boldsymbol{I}]\begin{bmatrix}\boldsymbol{w}\\ \boldsymbol{\epsilon}\end{bmatrix}$$

Note: We assume there are only a few outliers, so $\boldsymbol{\epsilon}$ contains only a few large values

OMP for Outlier Detection – Inputs

- Data
  - Given in the form of a data matrix $\boldsymbol{X}$
  - And targets $\boldsymbol{y}$
- Desired sparsity k of our solution
- Potentially also a desired threshold (more on this later)

OMP for Outlier Detection – Outputs

- A set of indices in our data that correspond to outlier data points
  - We'll call this F

The Algorithm: Initialization

Residual r

Matrix A

Outlier Set F

OMP: Initial Values

- **Residual:** difference of target and predicted value (using OLS)
  - $r = y - X(X^T X)^{-1} X^T y$
- **A:** matrix that is updated after each iteration based on the standard basis vector we have considered
  - $A = X$
- **Outlier Set:** We haven't found any outlier data points yet, so this will be initialized to the empty set
- We also set our index j to be 1 initially, and add 1 at each iteration

OMP: The Algorithm So Far

Algorithm 2: OMP to Detect Outliers

Input: Data matrix X, Predictions $\boldsymbol{y}$, Sparsity k, Residual Threshold ϵ

Output: Set of data indices F where the outliers are

$\boldsymbol{r} = \boldsymbol{y} - X(X^T X)^{-1} X^T \boldsymbol{y}$, j = 1, A = X, F = ∅;

while **do**

  j = j + 1;

end

return F

Identifying an index

- We'd like to pinpoint the index i with the largest error value (captured by our residual)
- How would we do this?
- Check for understanding: what should replace the red question mark below?

$$i = \underset{i}{\operatorname{argmax}} \left| \langle \boldsymbol{r}, ? \rangle \right|$$

Identifying an index

- We'd like to pinpoint the index i with the largest error value (captured by our residual)
- How would we do this?
- **We take the inner product of the residual with the i'th standard basis vector**

$$i = \underset{i}{\operatorname{argmax}} \left| \langle \boldsymbol{r}, \boldsymbol{e}_i \rangle \right|$$

$$i = \underset{i}{\operatorname{argmax}} \left| \langle \boldsymbol{r}, \boldsymbol{e}_i \rangle \right| = \underset{i}{\operatorname{argmax}} \left| \boldsymbol{r}[i] \right|$$

This simplifies to the i'th index of the residual, which we add to our set of outlier indices

OMP: The Algorithm So Far

Algorithm 2: OMP to Detect Outliers

Input: Data matrix X, Predictions $\boldsymbol{y}$, Sparsity k, Residual Threshold ϵ

Output: Set of data indices F where the outliers are

$\boldsymbol{r} = \boldsymbol{y} - X(X^T X)^{-1} X^T \boldsymbol{y}$, j = 1, A = X, F = ∅;

while $j \leq k$ *and* $||\boldsymbol{r}|| \geq \epsilon$ **do**

  i = $\text{argmax}_i\, |\boldsymbol{r}[i]|$;

  $F = F \bigcup \{i\}$;

  j = j + 1;

end

return F

Augmenting A

- We've pinpointed the i'th standard basis vector to have the maximum inner product with our residual
- So, we augment A by appending the i'th standard basis vector as follows:

$$A = [A|e_i];$$

Recomputing the Residual

- The next step to generate a new value of the residual, taking into account the fact that we've identified one new outlier data point
- To do this, we first perform OLS to generate a new prediction with our augmented version of A
  - Define $\boldsymbol{x} = \left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\boldsymbol{A}^T\boldsymbol{y}$
- Our updated prediction of the targets is $\boldsymbol{Ax}$

Recomputing the Residual

- The next step to generate a new value of the residual, taking into account the fact that we've identified one new outlier data point
- To do this, we first perform OLS to generate a new prediction with our augmented version of A

$$\text{Define } \boldsymbol{x} = \left(\boldsymbol{A}^{T}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{T}\boldsymbol{y}$$

- Our updated prediction of the targets is $\boldsymbol{Ax}$
- We then redefine our residual exactly as we did in the initial step, as the difference between the target vector and our prediction

$$\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{Ax}$$

OMP: The Algorithm So Far

Algorithm 2: OMP to Detect Outliers

Input: Data matrix X, Predictions $\boldsymbol{y}$, Sparsity k, Residual Threshold ϵ

Output: Set of data indices F where the outliers are

$\boldsymbol{r} = \boldsymbol{y} - X(X^T X)^{-1} X^T \boldsymbol{y}$, j = 1, A = X, F = $\emptyset$;

while **do**

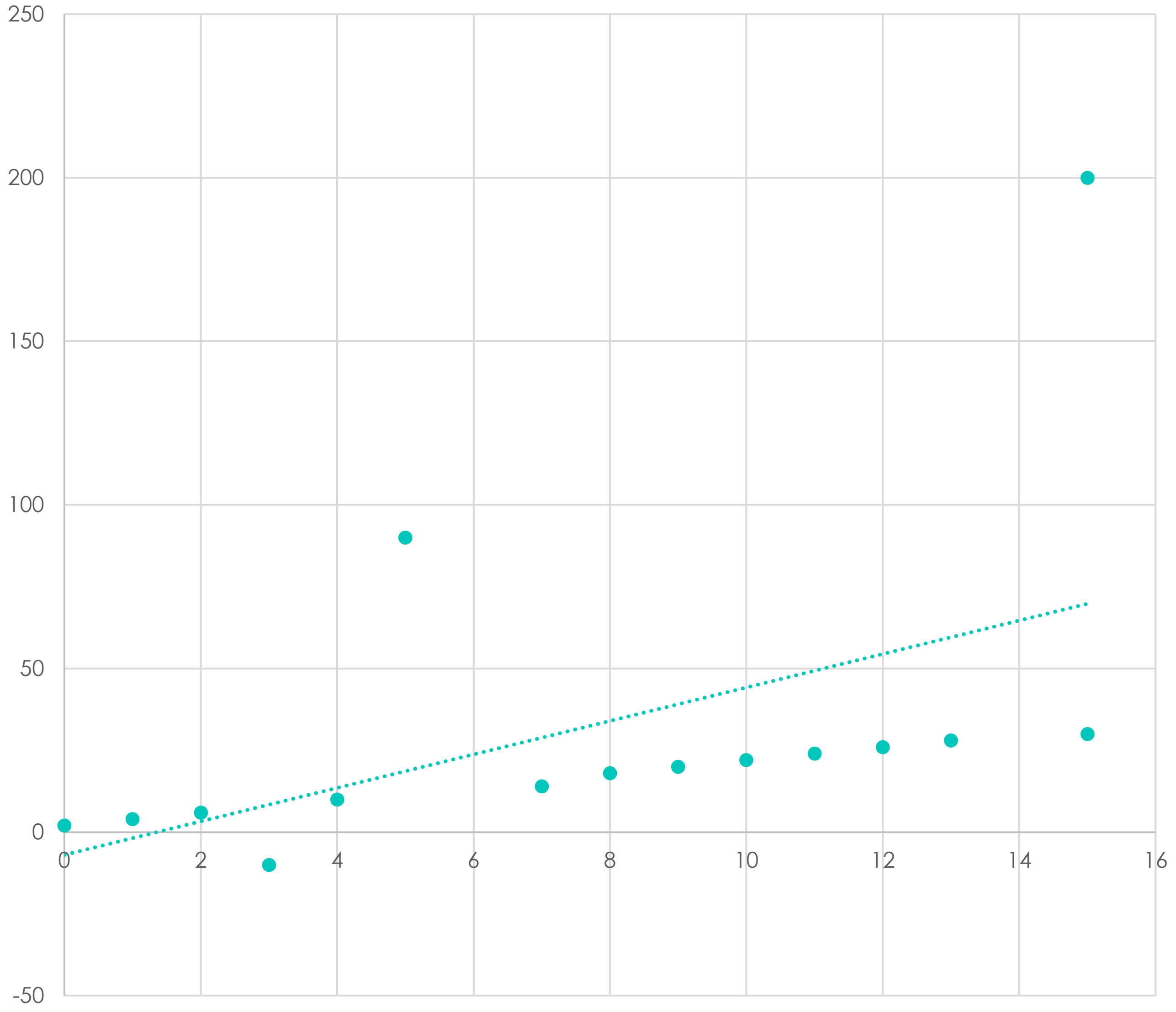
- i = $\text{argmax}_i \, |\boldsymbol{r}[i]|$;
- $F = F \bigcup \{i\}$;
- $A = [A|\boldsymbol{e_i}]$;
- $\boldsymbol{x} = (A^T A)^{-1} A^T \boldsymbol{y}$;
- $\boldsymbol{r} = \boldsymbol{y} - A\boldsymbol{x}$;
- j = j + 1;

end

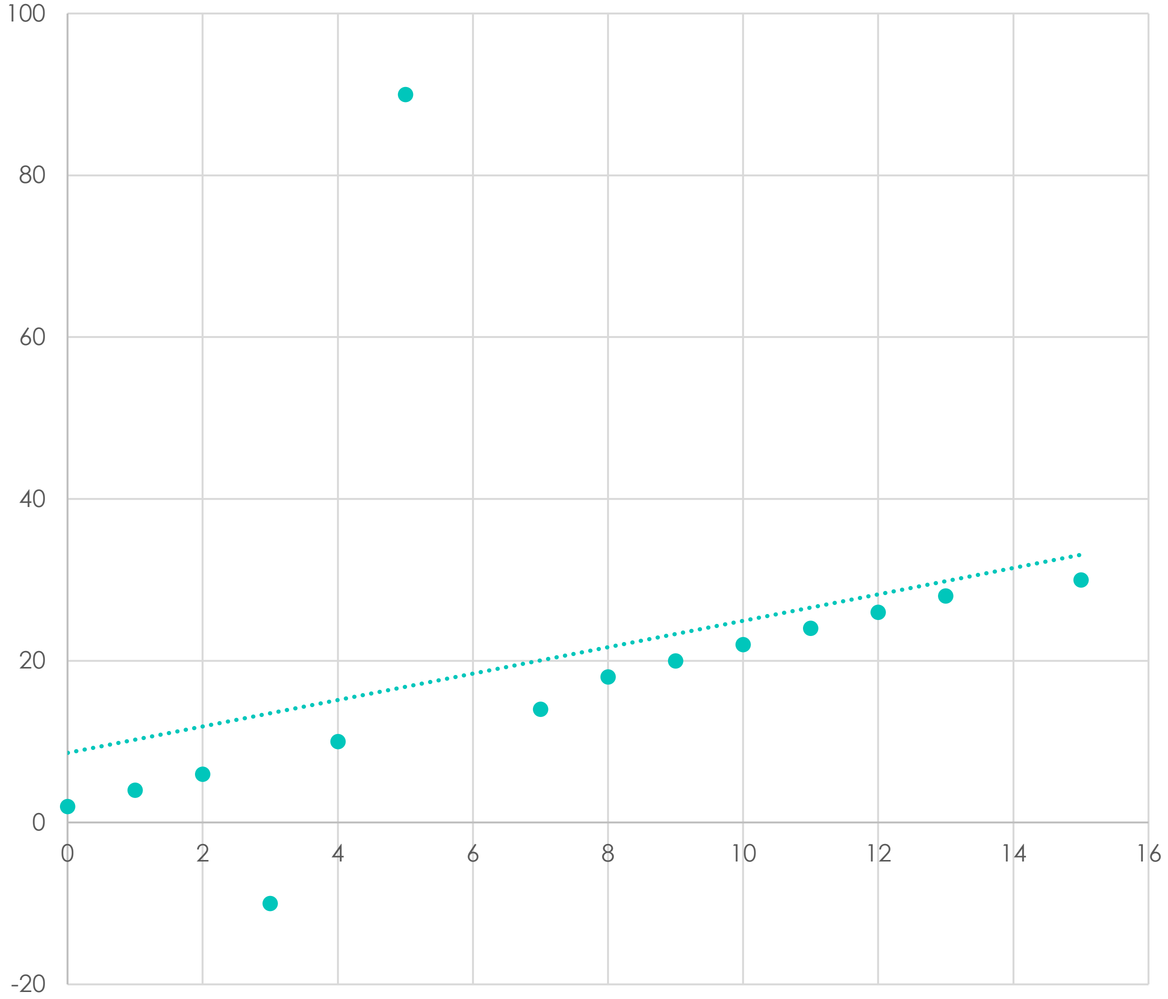
return F

OMP for Outlier Detection – Big Picture

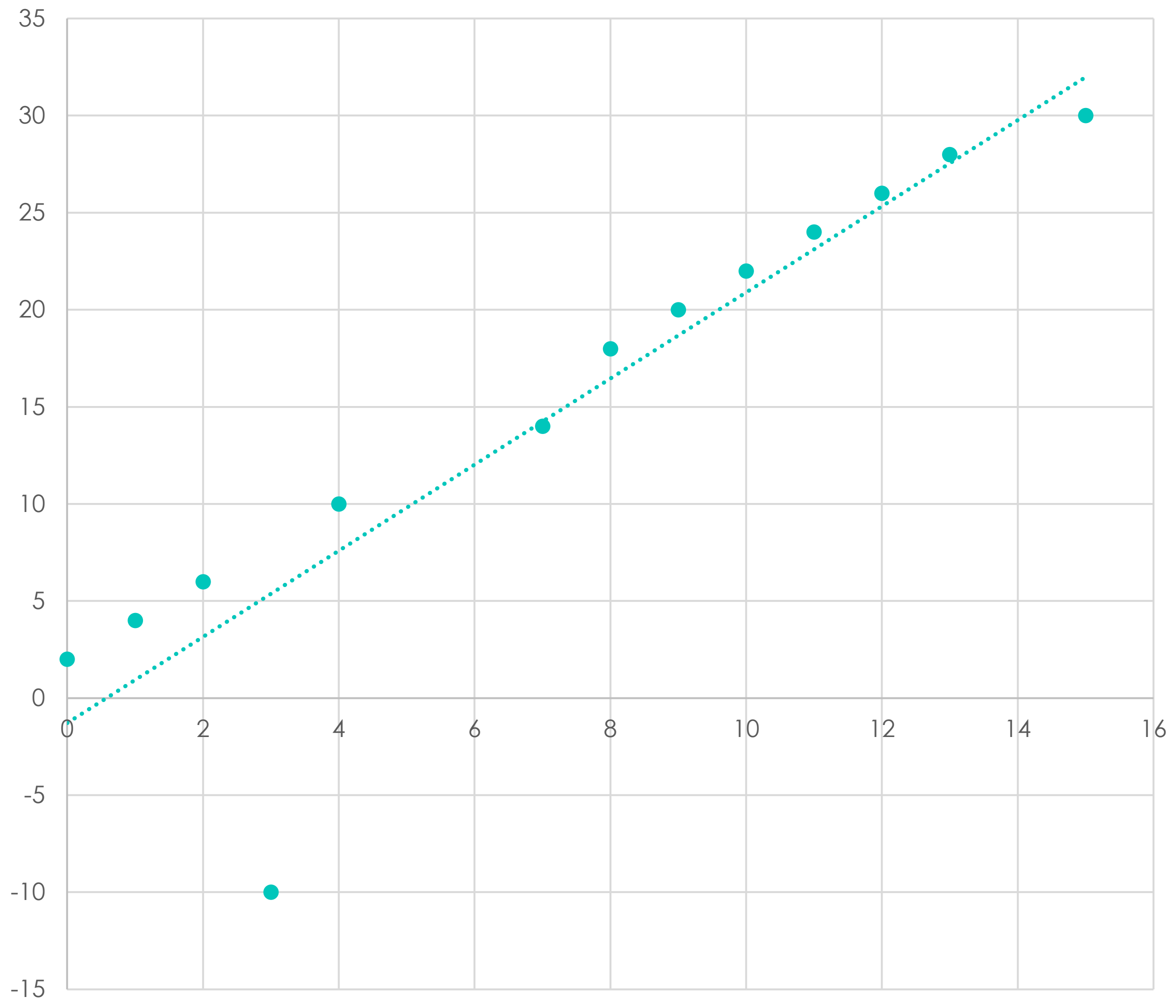
- Each iteration pinpoints one outlier to be removed
- Goal: finding the vectors in a given basis (e.g. the standard basis consisting of vectors e_i) that can best capture the pattern in our data, while excluding those directions that increase error in our predictions



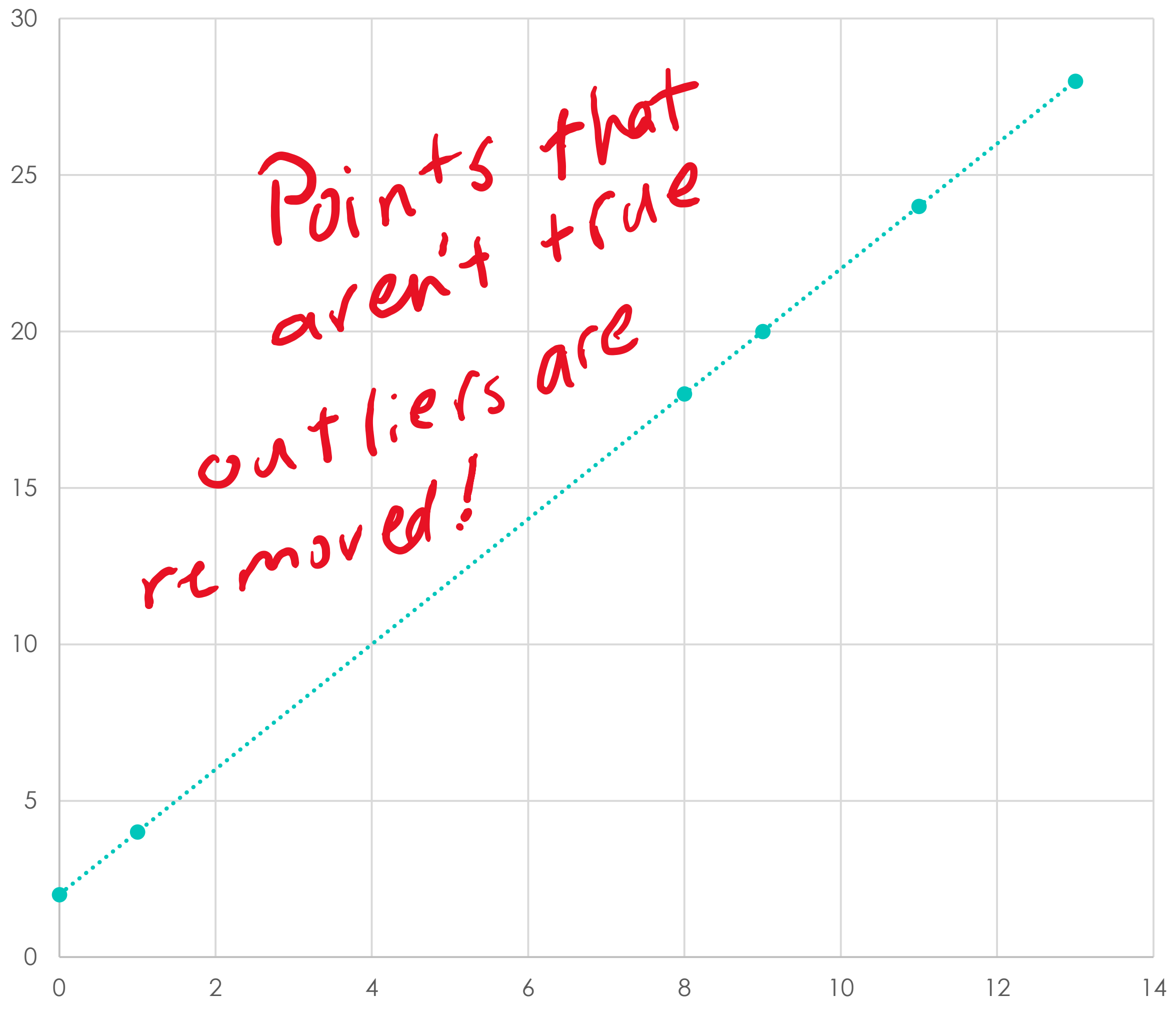
OMP for Outlier Detection and Removal - Visualization



OMP for Outlier Detection and Removal - Visualization



OMP for Outlier Detection and Removal - Visualization



After Several Iterations…

Question

How do we know when to stop running Orthogonal Matching Pursuit for outlier detection?

Stopping Conditions

- Option 1: Stop after a certain number of outliers are removed
  - This is especially helpful when we have an idea of how many outliers exist in our data
  - How does this affect our algorithm?
    - For k outliers, iterate until $j \leq k$

Stopping Conditions

- Option 1: Stop after a certain number of outliers are removed
  - This is especially helpful when we have an idea of how many outliers exist in our data
  - How does this affect our algorithm?
    - For k outliers, iterate until $j \leq k$
- Option 2: stop once we've reached a certain residual threshold
  - Stop iterating when the norm of our residual r falls below a predefined value ϵ
  - How does this affect our algorithm?
    - Iterate while $||r|| \geq \epsilon$

Note: the values of k and epsilon are set **by us**, and are not a quantity determined by the OMP algorithm.

This means we can tune them!

OMP for Outlier Detection – Putting it all together

Algorithm 2: OMP to Detect Outliers

Input: Data matrix X, Predictions $\boldsymbol{y}$, Sparsity k, Residual Threshold ϵ
Output: Set of data indices F where the outliers are
$\boldsymbol{r} = \boldsymbol{y} - X(X^T X)^{-1} X^T \boldsymbol{y}$, j = 1, A = X, F = $\emptyset$;
while $j \leq k$ *and* $||\boldsymbol{r}|| \geq \epsilon$ **do**
  i = $\operatorname{argmax}_i |\boldsymbol{r}[i]|$;
  $F = F \bigcup \{i\}$;
  $A = [A|\boldsymbol{e_i}]$;
  $\boldsymbol{x} = (A^T A)^{-1} A^T \boldsymbol{y}$;
  $\boldsymbol{r} = \boldsymbol{y} - A\boldsymbol{x}$;
  j = j + 1;
end
return F