KIOS Research and Innovation Center of Excellence

MSc on Intelligent Critical Infrastructure Systems

Machine Learning

Lecture 7 and 8

Marios Polycarpou

Director, KIOS Research and Innovation Center of Excellence

Professor, Electrical and Computer Engineering

University of Cyprus

University of Cyprus

Imperial College London

funded by:

Outline

- Supervised Learning (continuation)
  - Learning algorithm selection
  - Underfitting and overfitting
  - Regularization
- Online Learning

Learning algorithm selection

- Number of features and examples
- In-memory vs. out-of-memory
- Categorical vs. Numerical features
- Nonlinearity of data
- Training speed and prediction speed
- Explainability (Explainable AI)

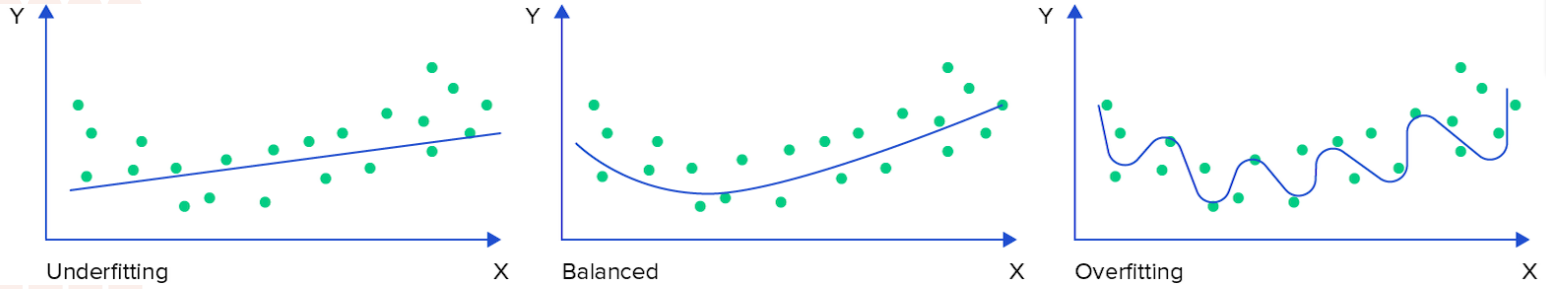
→ Possible to try various machine learning algorithms and select one by testing on validation test.

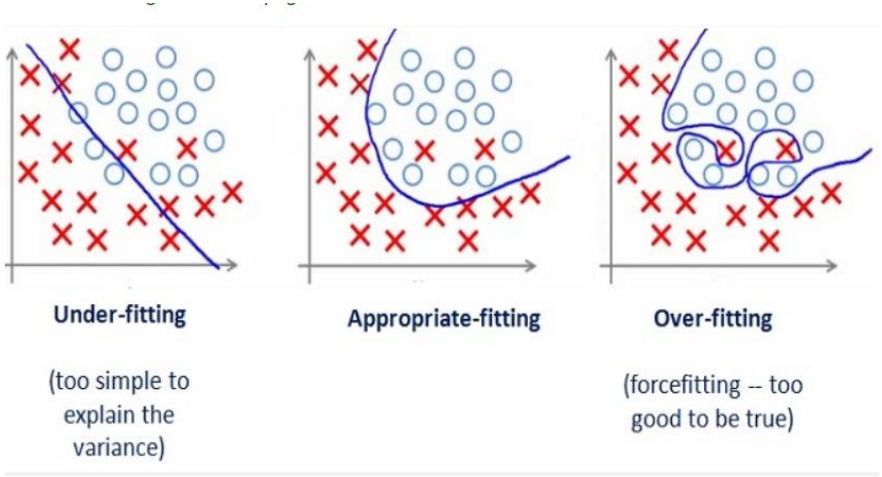
- **Training set**
- **Validation set**
- **Test set**

Underfitting and overfitting

Undefitting → high bias

Overfitting → high variance





Main reasons for underfitting

- Model is too simple for the data
- Features not sufficiently suitable to describe the underlying correlations

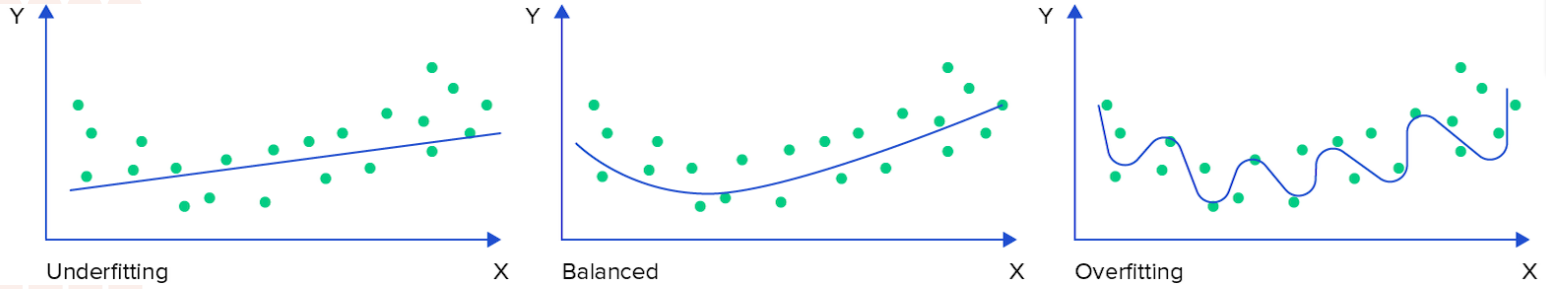
Main reasons for overfitting

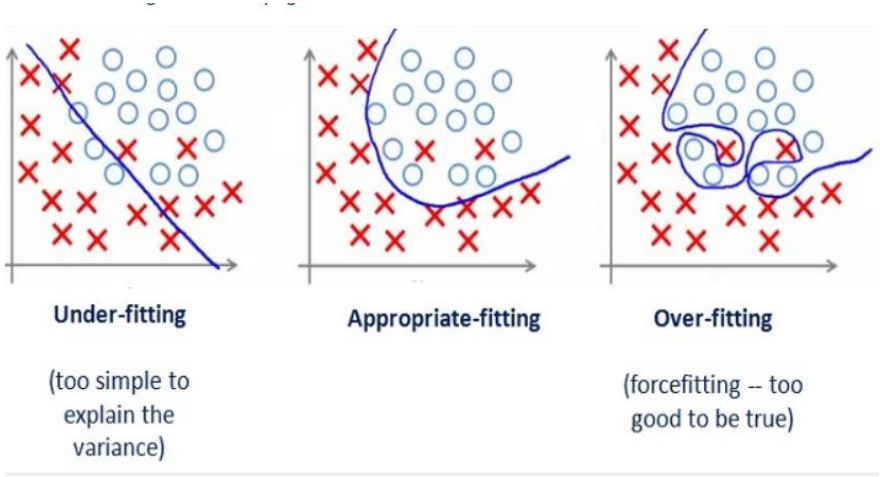
- Model is too complex for the data
- Too many features but small number of training examples

Underfitting and overfitting

Undefitting → high bias

Overfitting → high variance





How to address the overfitting problem

- Try a simpler model (what does that mean?)
- Reduce the dimensionality/number of features (dimensionality reduction methods)
- Add more training data
- Regularize the learning model

Regularization

$$J(\theta) = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{f}(x_n;\theta) - y_n\right)^2 \Rightarrow J_R(\theta) = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{f}(x_n;\theta) - y_n\right)^2 + \lambda\|\theta\|^2$$

$$J_R(\theta) = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{f}(x_n;\theta) - y_n\right)^2 + \lambda\sum_{i=1}^{M}|\theta_i|$$

L1 Regularization
(Lasso regularization)

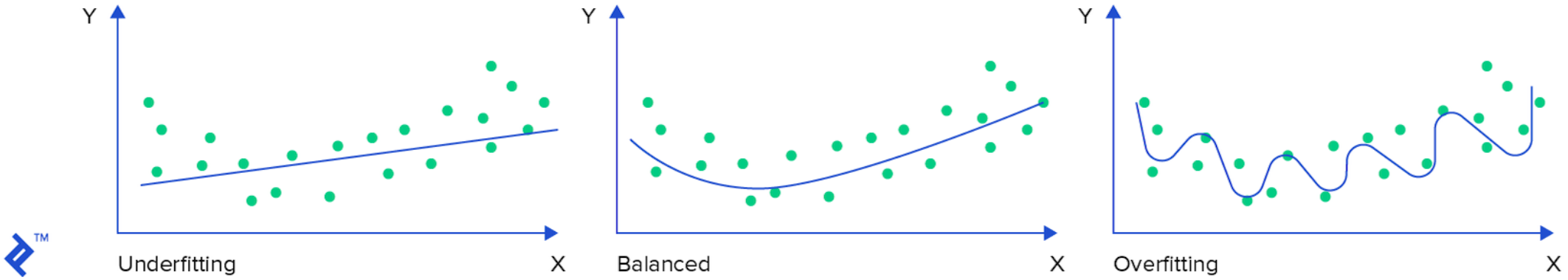
$$J_R(\theta) = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{f}(x_n;\theta) - y_n\right)^2 + \lambda\sum_{i=1}^{M}\theta_i^2$$

L2 Regularization
(Ridge regularization)
(Tikhonov regularization)

L1 and L2 Regularization

- λ ıs a hyperparameter, which can be varied to balance between reducing bias and variance (reducing training error and enhancing generalization)
- L1 regularization produces a sparse model – sets most of the parameters to zero. Therefore, L1 is similar to performing feature selection. Useful for explainability of machine learning algorithm.
- From a performance viewpoint, L2 typically gives better results. Also, has the advantage of being differentiable, which is important if using gradient descent to minimize the cost function.

Intuition behind regularization



$$\hat{f}(x;\theta) = \theta_0 + \theta_1 x$$

$$\hat{f}(x;\theta) = \theta_0 + \theta_1 x + \theta_2 x^2$$

$$\hat{f}(x;\theta) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \theta_5 x^5$$

$$J_R(\theta) = \frac{1}{N}\sum_{n=1}^{N}\left(\hat{f}(x_n;\theta) - y_n\right)^2 + \lambda\theta_3^2 + \lambda\theta_4^2 + \lambda\theta_5^2$$

As λ becomes large (say, λ=100), then the optimal $\theta_3 \approx \theta_4 \approx \theta_5 \approx 0$

which reduces overfitting

Gradient Descent with Regularization

Linearly parametrized approximation models

$$\hat{f}(x) = \sum_{i=1}^{M} \theta_i \phi_i(x) = \theta^T \phi(x) = \phi(x)^T \theta \qquad J(\theta) = \frac{1}{N} \sum_{i=1}^{N} (\theta^T \phi(x) - y_i)^2 + \lambda \|\theta\|^2$$

$$\theta(k+1) = \theta(k) - \alpha \left(\frac{2}{N} \left(\theta^T(k) \phi(x(k)) - y(k) \right) \phi(x(k) + 2\lambda\theta(k) \right)$$

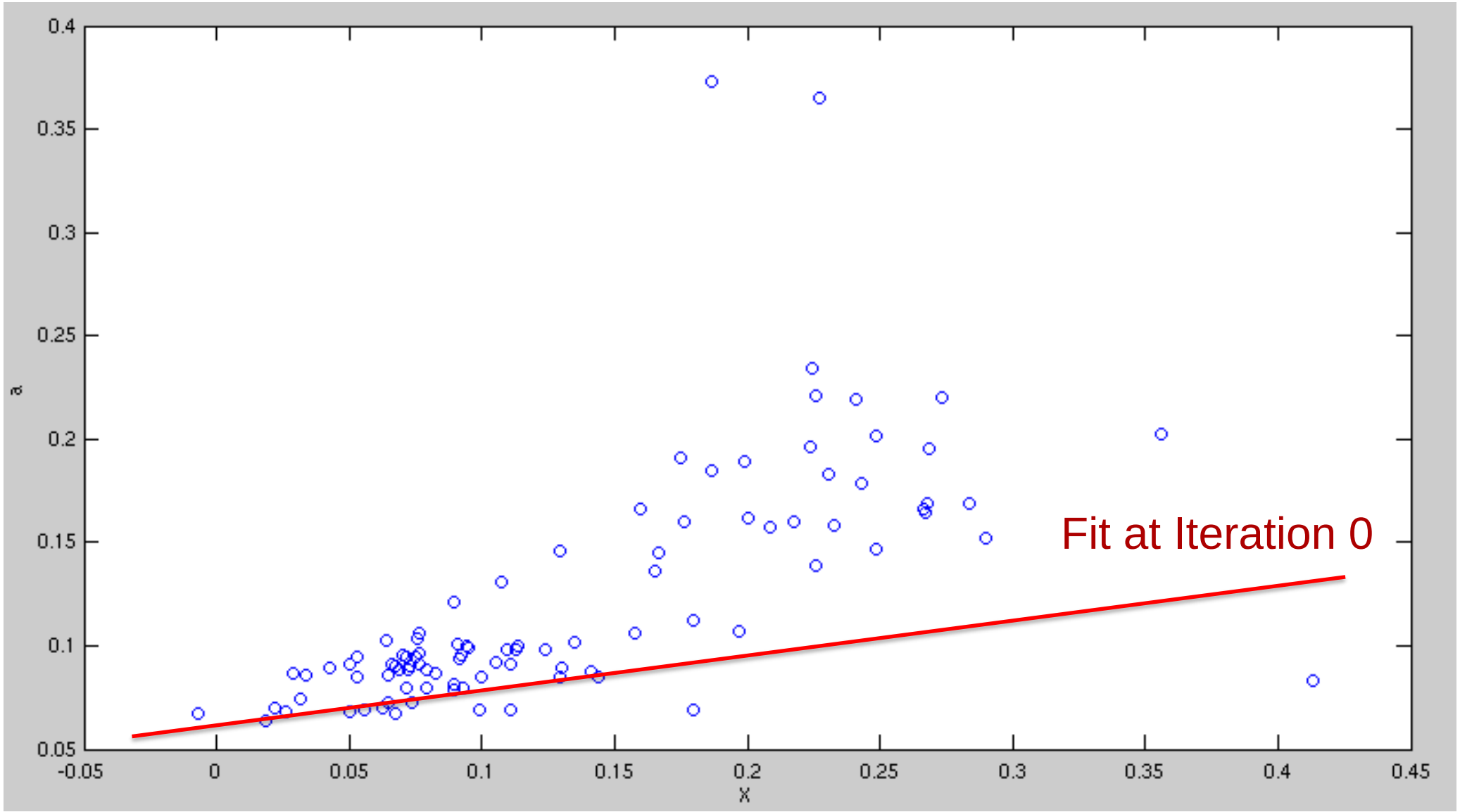
$$\theta(k+1) = (1 - 2\alpha\lambda)\theta(k) - \alpha \left(\frac{2}{N} \left(\theta^T(k) \phi(x(k)) - y(k) \right) \phi(x(k) \right)$$

$$\theta(k+1) = \left(1 - \tilde{\lambda}\right)\theta(k) - \tilde{\alpha} \left(\left(\theta^T(k) \phi(x(k)) - y(k) \right) \phi(x(k) \right)$$

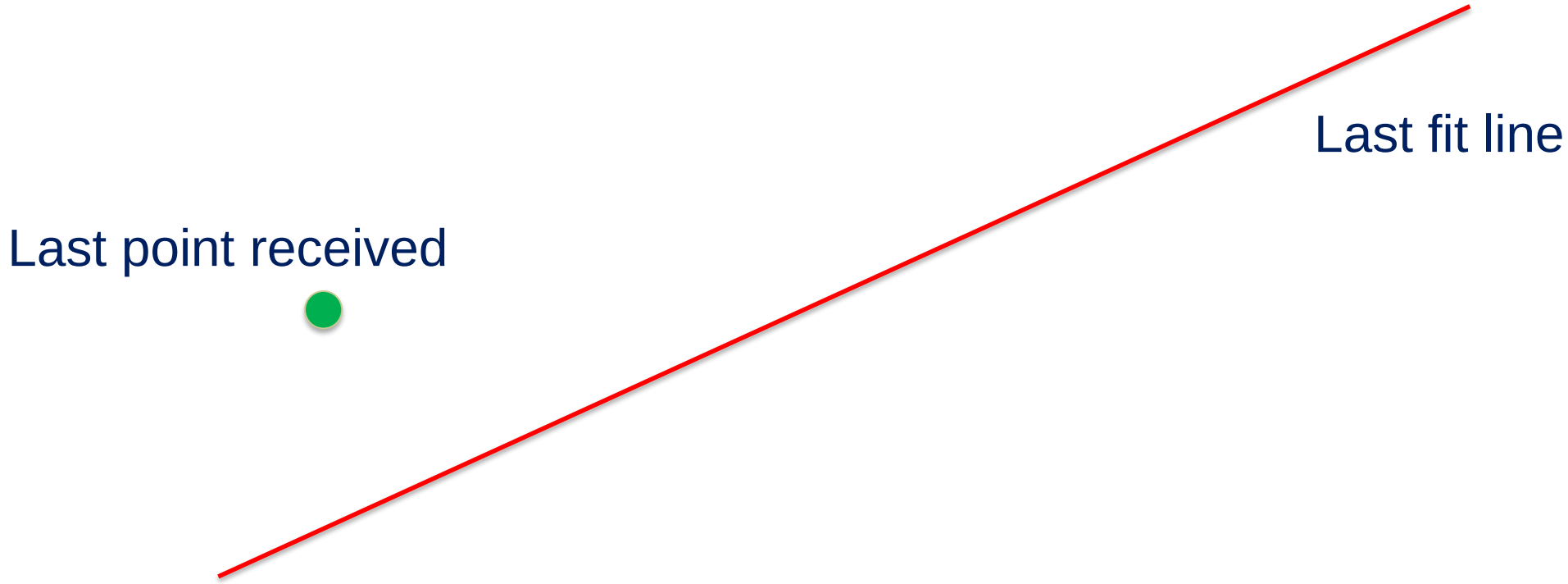
Online Learning

- In offline learning, data is collected for some time (batch) and then machine learning algorithms are applied on the batch data.
- In online learning, training occurs in consecutive rounds. At the beginning of each round, the algorithm is presented with an input sample, based on which it makes a prediction. Based on the difference between the prediction and the desired/true output, the model is adapted for subsequent rounds.
- Online learning doesn't necessarily mean streaming data, but usually it is applied to streaming data. Sometimes, even called *streaming learning*.
- Slow/fast learning vs. online/offline learning

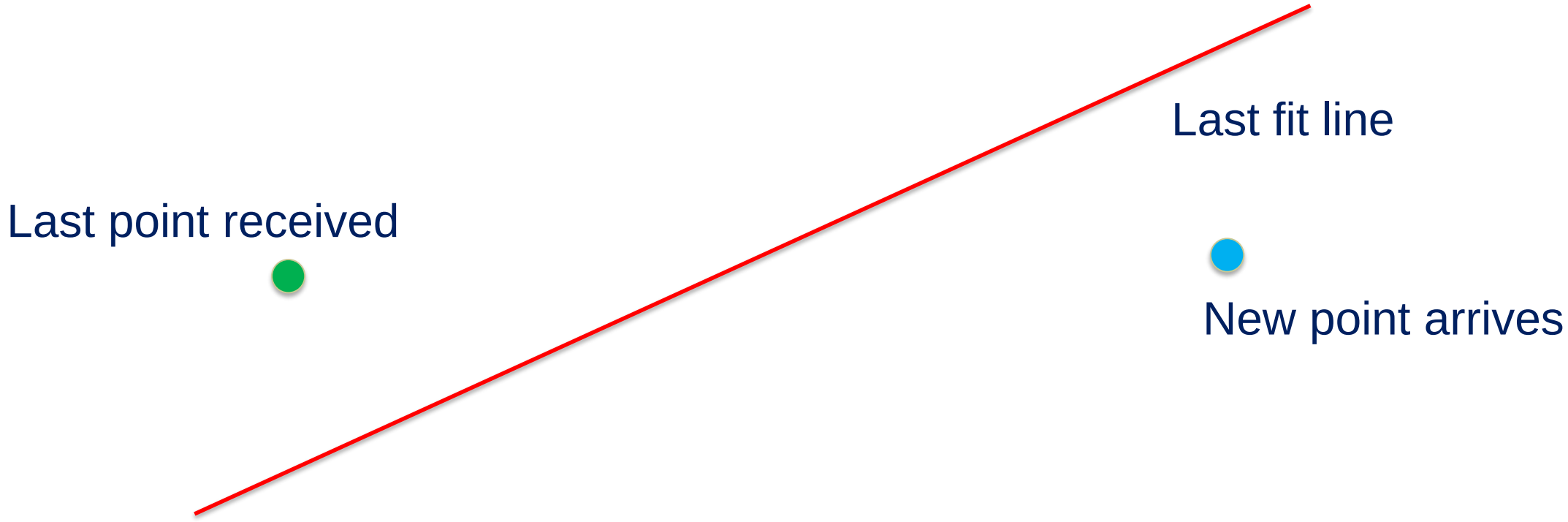
Offline Linear Regression



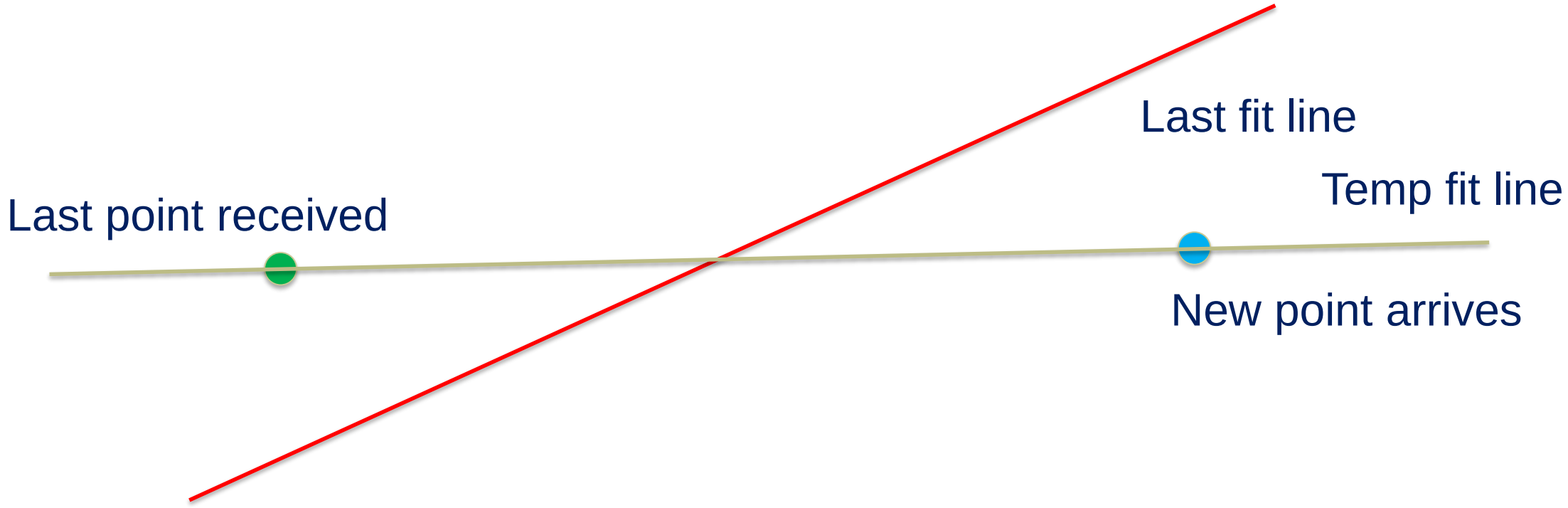
Online Linear Regression



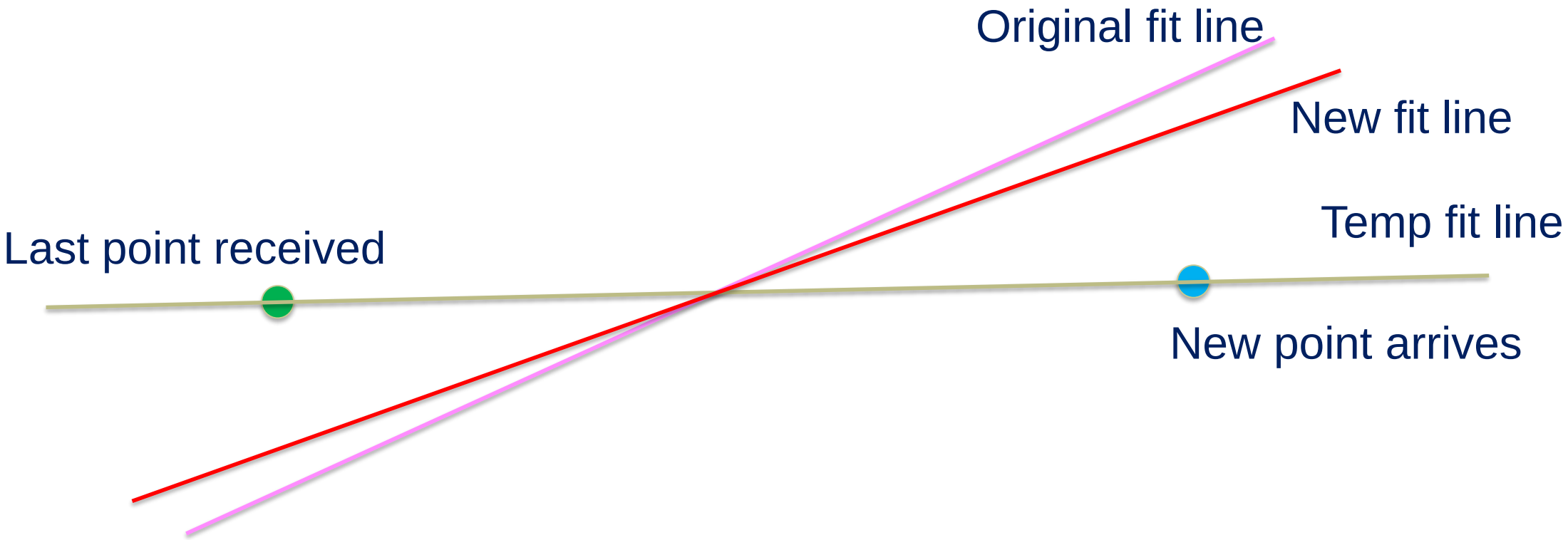
Online Linear Regression



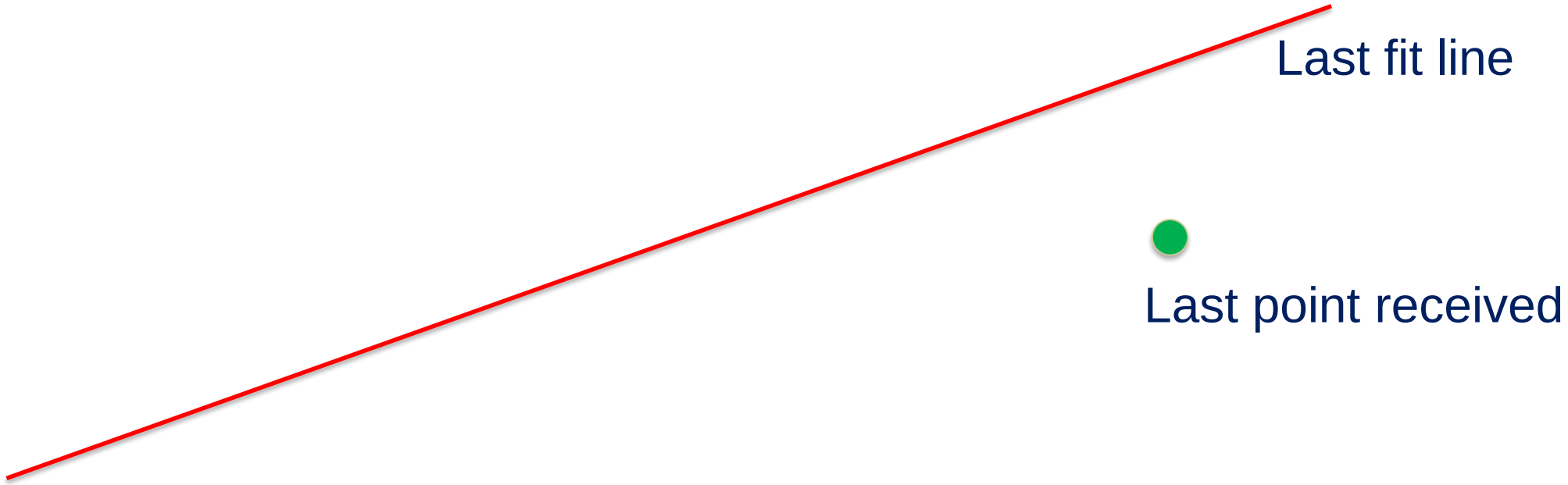
Online Linear Regression



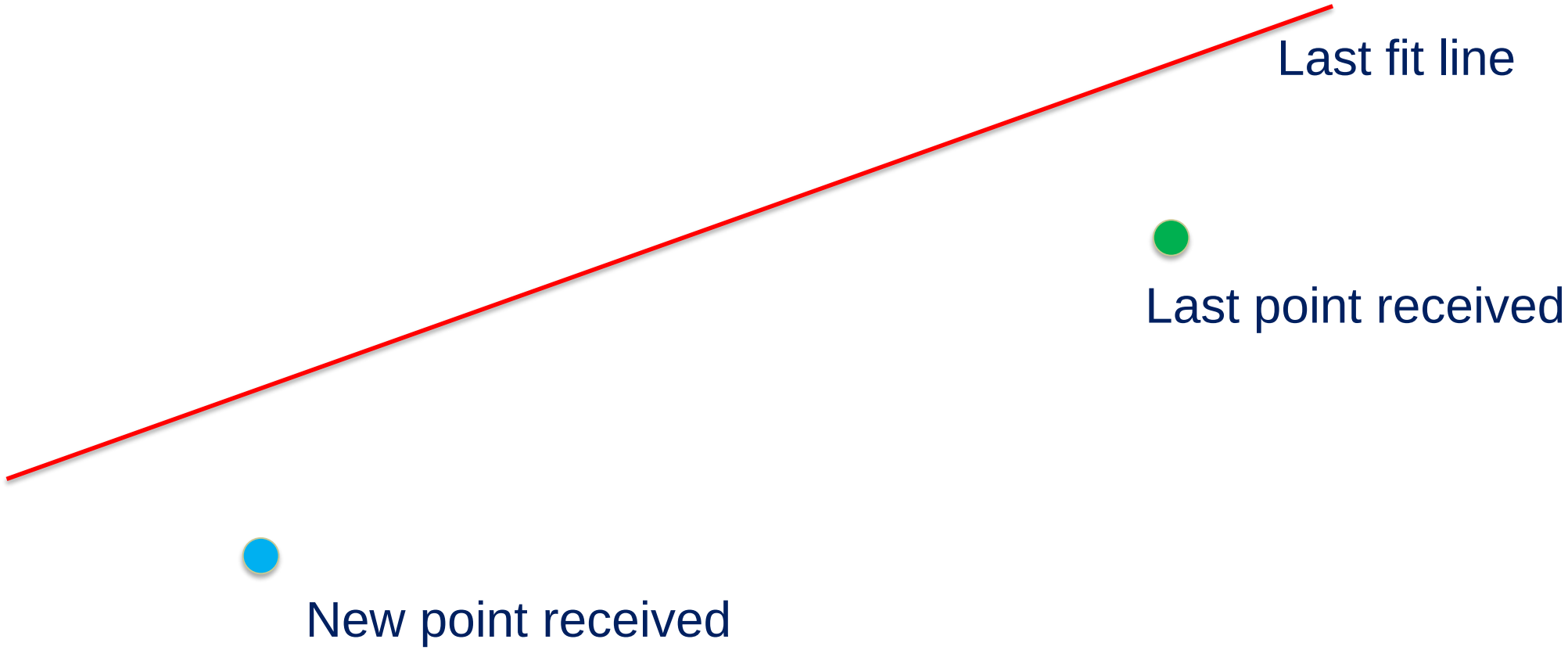
Online Linear Regression



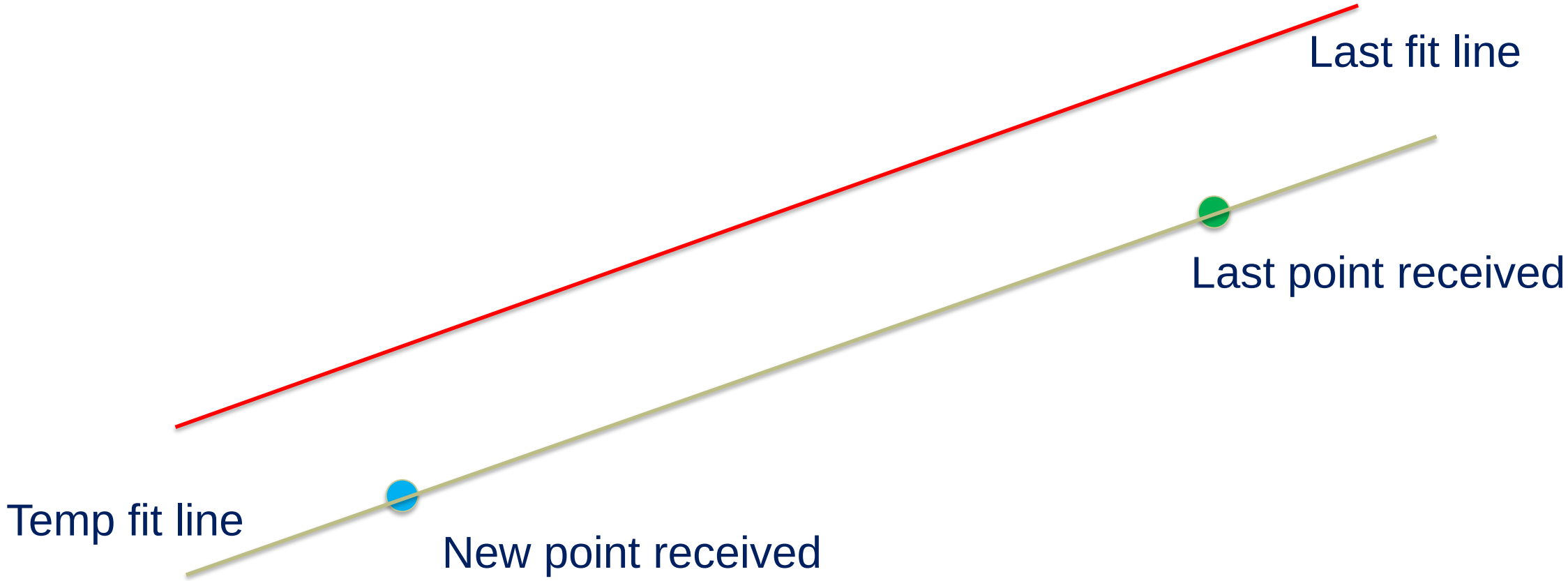
Online Linear Regression



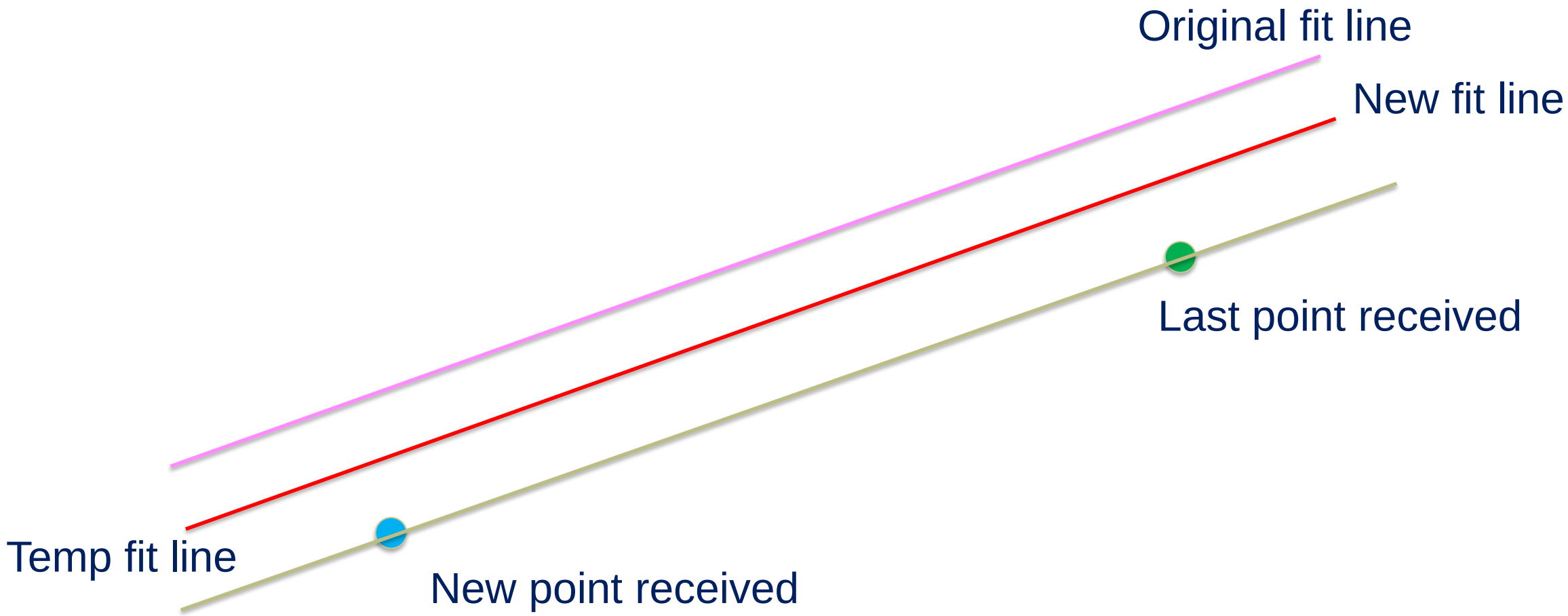
Online Linear Regression



Online Linear Regression

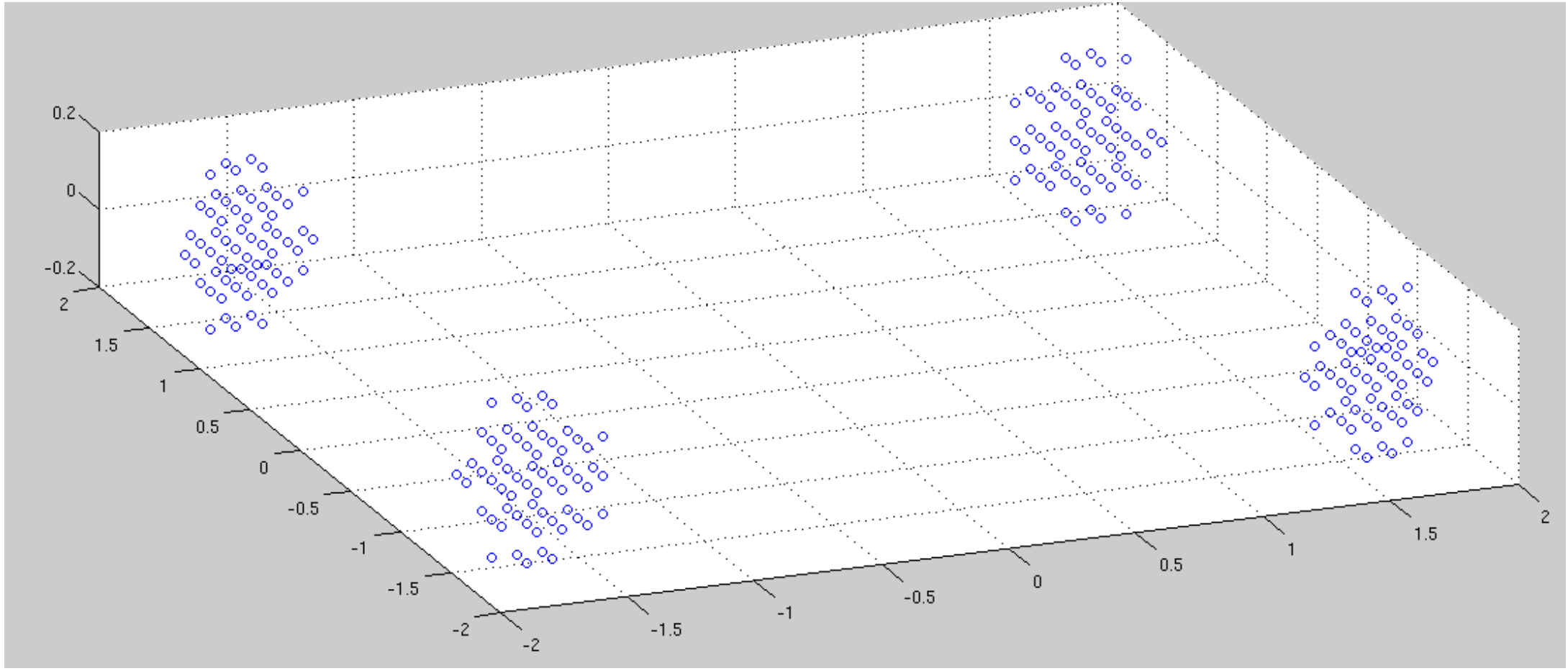


Online Linear Regression



Offline Clustering

Step 0

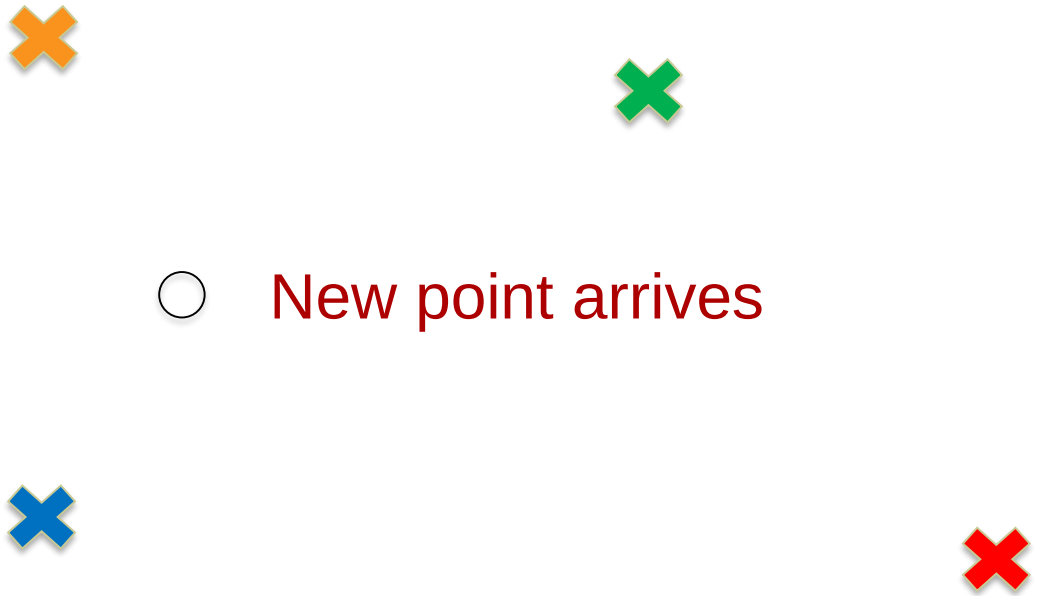


Online Clustering

Step k-1

Online Clustering

Step k



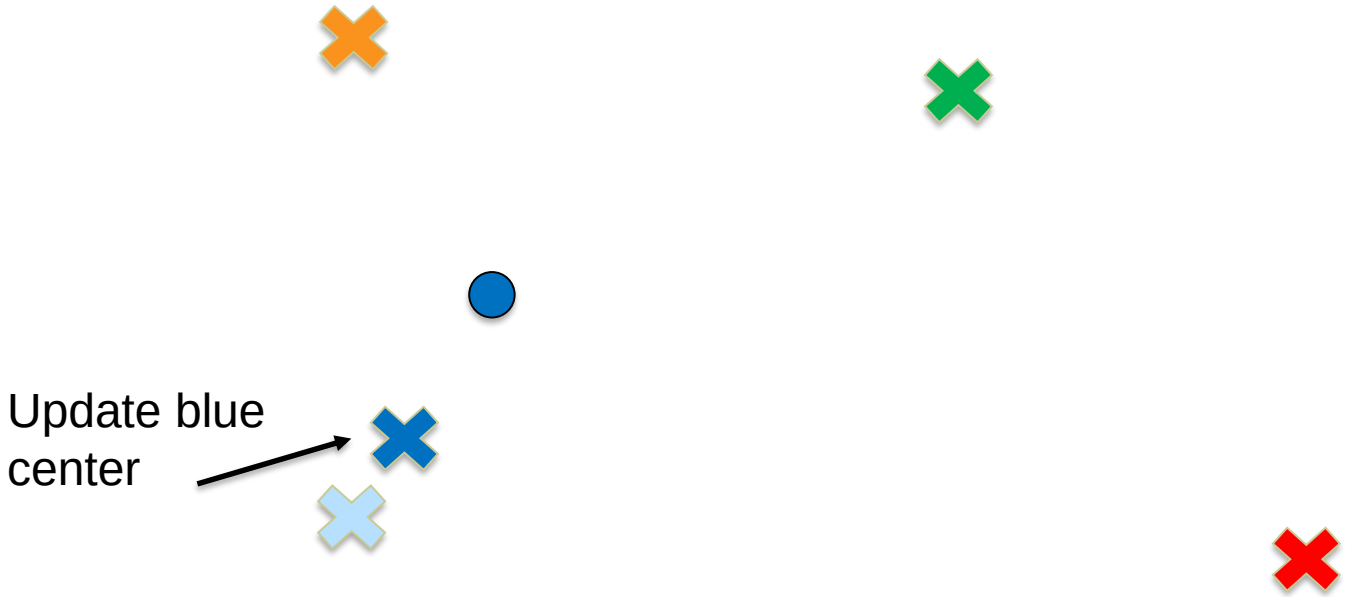
New point arrives

Online Clustering

Step k

Online Clustering

Step k



Online Learning – key concepts

For k = 1, 2, ….

receive question (input) $x_k \in X$

predict answer (output) $\hat{y}_k \in Y$

receive true answer (output) $y_k \in Y$

suffer loss $L(\hat{y}_k - y_k)$

update the model for generating predictions for next trials

Online Learning – key concepts

A key objective of online learning algorithms is to minimize the **regret**.

The **regret** is the difference between the performance of the online algorithm and an ideal algorithm that has been able to train on the whole data seen so far, in batch fashion; i.e., an online machine learning algorithm is trying to perform as closely as possible to an ideal corresponding offline algorithm.

Online Learning – Application Examples

- Fraud detection
- Spam detection
- Financial portfolio selection
- Online ad placement
- Online web ranking
- Real-time monitoring
- Navigation and control

Online Learning – Stationary / Time-Varying Targets

The target function that we are trying to learn maybe stationary or dynamic.

- **Stationary Targets**. The target function we are trying to learn does not change over time, but it is unknown (or uncertain) and it may be stochastic.
- **Non-stationary (time-varying) Targets**. The target function we are trying to learn not only it is unknown, but it is changing over time. It may even be adapting to our model (for example, in an adversarial manner).

Online Learning in dynamical systems

$$y(k+1) = f\left(y(k), y(k-1), \ldots y(k-n_y, u(k), \ldots u(k-n_u\right)$$

$k \in \mathbb{Z}^+ \qquad k = 0, 1, 2, \ldots\ldots$

$f : \mathbb{R}^{n_y+1} \times \mathbb{R}^{n_u+1} \rightarrow \mathbb{R}$

f is unknown (or partially unknown)

Let $z(k) = \left[y(k), y(k-1), \ldots y(k-n_y, u(k), \ldots u(k-n_u \right]$

$\Rightarrow y(k+1) = f(z(k))$ $(z(k)$ is measurable)

Online Learning in dynamical systems

$$y(k+1) = f(z(k))$$
$$\hat{y}(k+1) = \hat{f}(z(k); \hat{\theta}(k))$$

← **Prediction/Identification Model**

Adaptive Laws:

$$\hat{\theta}(k+1) = \hat{\theta}(k) + \alpha_0 e(k+1)\xi(k)$$

$$\hat{\theta}(k+1) = \hat{\theta}(k) + \frac{\gamma_0 e(k+1)}{\beta_0 + |\xi(k)|^2}\xi(k)$$

Normalized Gradient Descent

$\alpha_0 > 0$ Step size

$0 < \gamma_0 < 2$ Learning rate

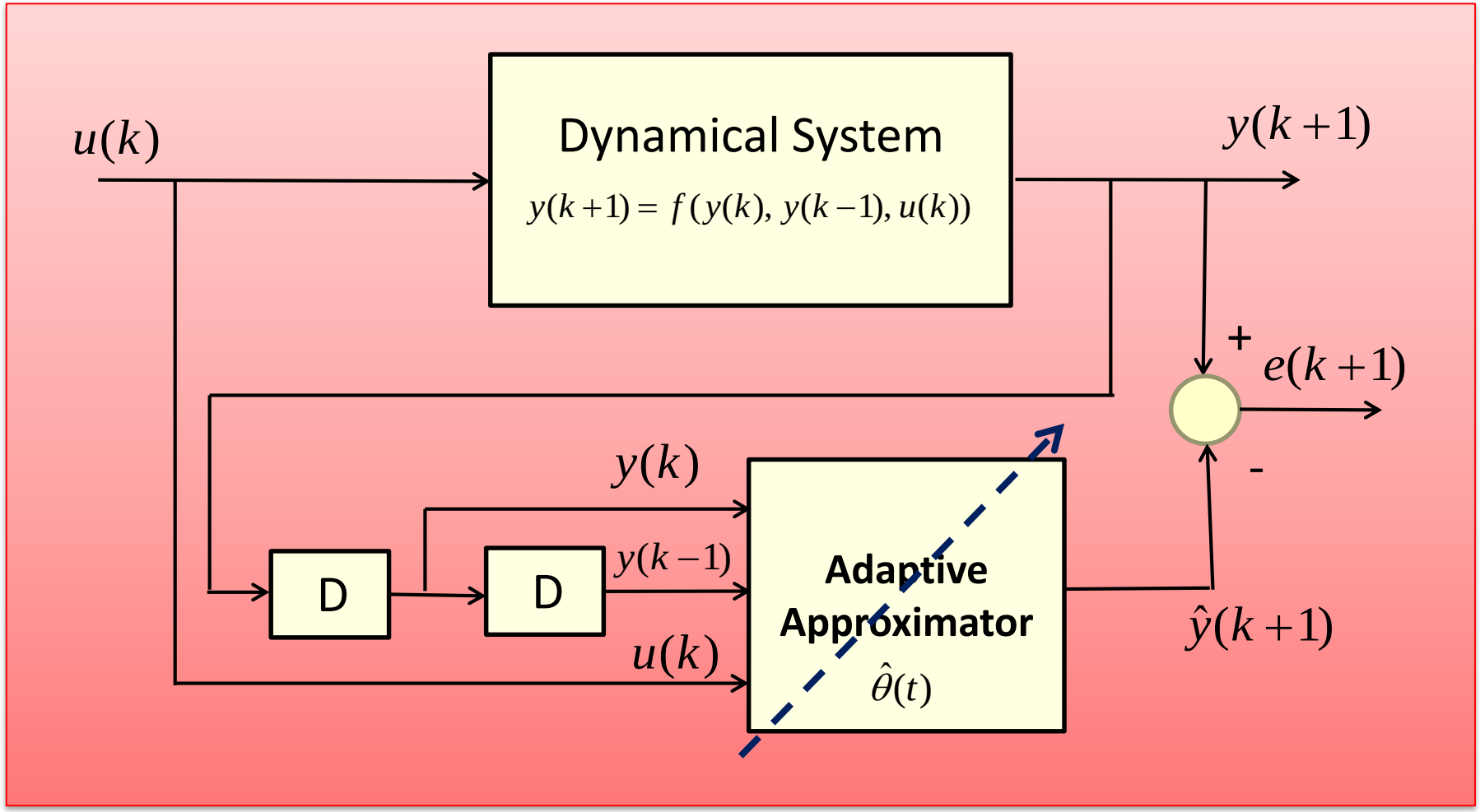
$\beta_0 > 0$ Small design constant

$\xi(k) = \frac{\partial \hat{f}}{\partial \hat{\theta}}\left(z(k), \hat{\theta}(k)\right)$ Network sensitivity function

$e(k) = y(k) - \hat{y}(k)$ Estimation error

Online Learning in dynamical systems

Example: $y(k+1) = f(y(k), y(k-1), u(k))$



Online Learning in dynamical systems

Example: $y(k+1) = f(y(k), y(k-1), u(k))$

Given $y(0), y(-1), \hat{\theta}(0), u(0)$

\begin {for} $k = 0, 1, 2, \ldots\ldots N$

1. $z(k) = \left[y(k), y(k-1), u(k) \right]$
2. $\xi(k) = \dfrac{\partial \hat{f}}{\partial \hat{\theta}} \left(z(k), \hat{\theta}(k) \right)$
3. $y(k+1) = f(z(k))$
4. $\hat{y}(k+1) = \hat{f}(z(k), \hat{\theta}(k))$
5. $\hat{\theta}(k+1) = \hat{\theta}(k) + \dfrac{\gamma_0 \left(y(k+1) - \hat{y}(k+1) \right)}{\beta_0 + \left| \xi(k) \right|^2} \xi(k)$

\end {for}