Development and application of disruption predictors for real time detection

J. Vega

EUROfusion Consortium, JET, Culham Science Centre, Abingdon, OX14 3DB, UK
Laboratorio Nacional de Fusión, CIEMAT, Madrid, Spain

Acknowledgements
This work was partially funded by the Spanish Ministry of Economy and Competitiveness under the Projects No ENE2015-64914-C3-1-R and ENE2015-64914-C3-2-R.

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

Outline

- Introduction
- Machine learning for automatic classifiers
- Disruption predictors as classification systems
- Combination of the temporal domain and the frequency domain of the signals for disruption prediction
- APODIS disruption predictor
  - Overview of Support Vector Machines
- Disruption predictors from scratch
  - APODIS based
  - Venn predictors
- Disruption predictors based on anomaly detection
  - Clustering based
- Summary

Introduction

- Experimental nuclear fusion devices like JET or ITER need to explore advanced operation scenarios to achieve high performance plasmas
  - Plasmas near operational limits can produce disruptions
- Avoidance methods
  - To maintain the plasma away from disruptive behaviours
- When a disruption is inevitable, mitigation actions are required
  - Reducing forces, alleviating heat loads during the thermal quench and avoiding runaway electrons

Introduction

- Existing techniques of avoidance/mitigation methods are
  - Injection of significant amount of gases through fast valves
    - M. Lehnen et al. Nuclear Fusion 51 (2011) 123010 (12pp)
    - M. Bakhtiari et al. Nuclear Fusion 51 (2011) 063007 (9pp)
  - Killer pellets
    - G. Pautasso et al. Nuclear Fusion, 36, 10 (1996) 1291-1297
    - N. Commaux et al. Nuclear Fusion 51 (2011) 103001 (9pp)
  - ECRH injection
    - B. Esposito et al. Phys. Rev. Lett. 100, 045006 (2008)
    - B. Esposito et al. Nuclear Fusion 51 (2011) 083051 (9pp)

Introduction

- A reliable real-time disruption predictor is a pre-requisite to any mitigation method
  - Reliability has to be understood in terms of
    - High success rate (disruptions correctly predicted)
    - Low false alarm rate (non-disruptive discharges predicted as disruptive)
    - Enough anticipation time
      - Premature alarms (warning times greater than reasonable anticipation times)
      - Tardy detections (alarms produced with a warning time less than the time required for mitigation actions)

- The objective of a disruption predictor is to recognize the presence of disruption precursors and to trigger an alarm ASAP

Introduction

- So far, plasma theoretical models do not cope satisfactorily with the prediction of disruptions
  - The lack of feasible disruption theory cannot stop the research on nuclear fusion and, therefore, alternative methods are used

- Simple real-time predictors trigger an alarm when the locked mode is above a user-defined threshold
  - However, this is not reliable enough
  - In all tokamaks, a common signal used for disruption prediction is the locked mode (LM) amplitude
  - When macroscopic instabilities start locking to the wall, the LM amplitude grows during the slowing down of the plasma rotation
  - During a running experiment, if the LM amplitude crosses a certain threshold, which is set-up prior to the discharges, an alarm is triggered
  - This threshold is selected in a manual way and it is chosen depending on the characteristics of the experimental programme: the LM threshold is set lower or higher depending on the potential danger of the possible disruptions

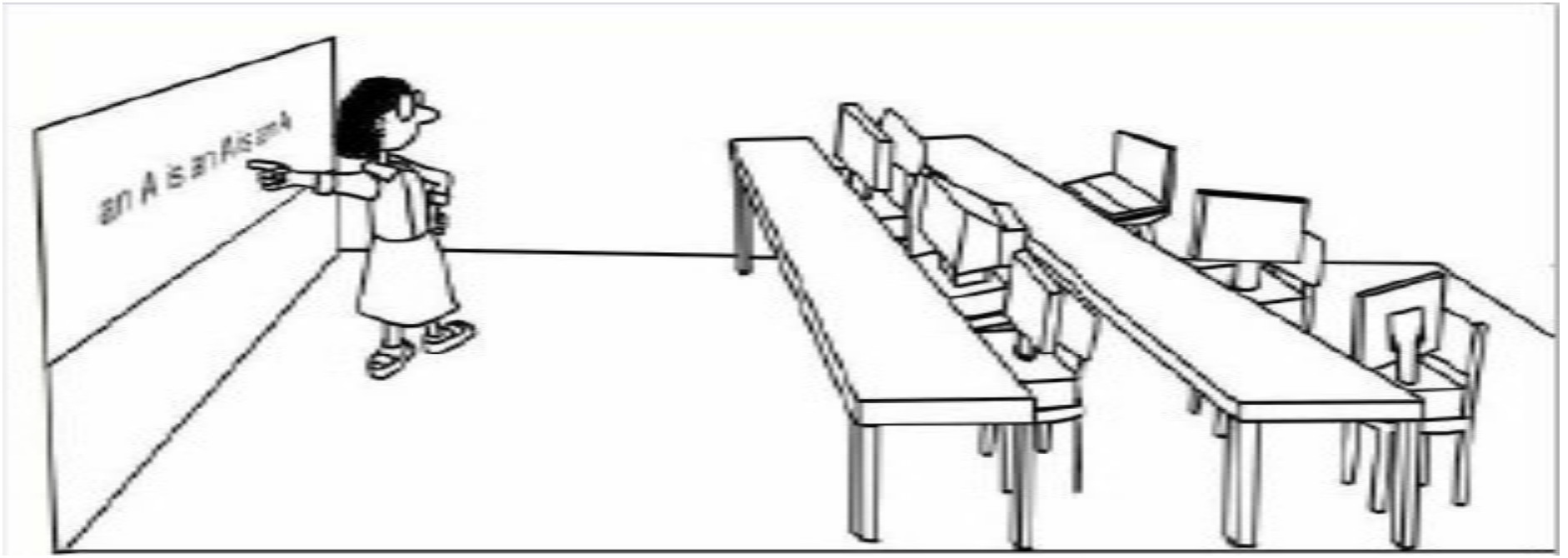
Introduction

- In general, disruption predictors are based on data-driven approaches and machine learning techniques
  - The objective is to recognise disruptive behaviours although the physics reasons are unknown

- This presentation describes three different kinds of disruption predictors
  - Results in JET are shown

Machine learning for automatic classifiers

Machine learning: concepts

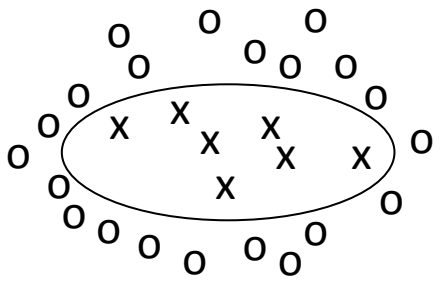


- Learning does not mean '*learning by heart*' (any computer can memorize)

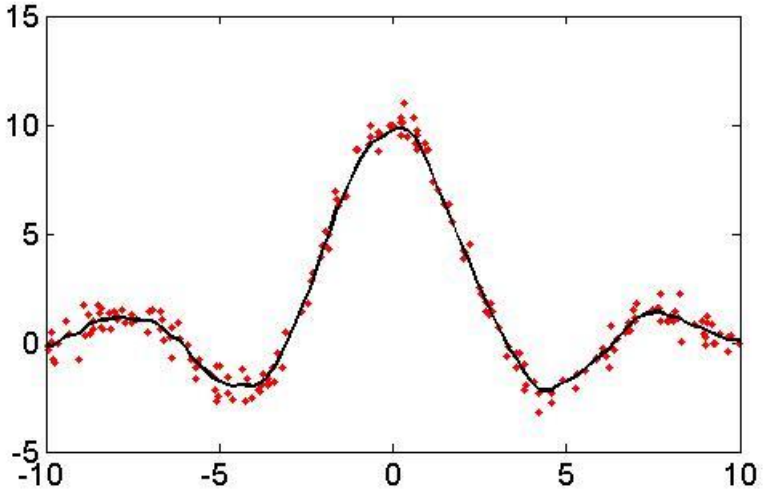
- Learning means '*generalization capability*': we learn with some examples and predict for other examples

Machine learning: concepts

- The learning problem is the problem of finding a desired dependence (function) using a limited number of observations (training data)
  - *Classification*: the function can represent the separation frontier between two classes
  - *Regression*: the function can provide a fit to the data



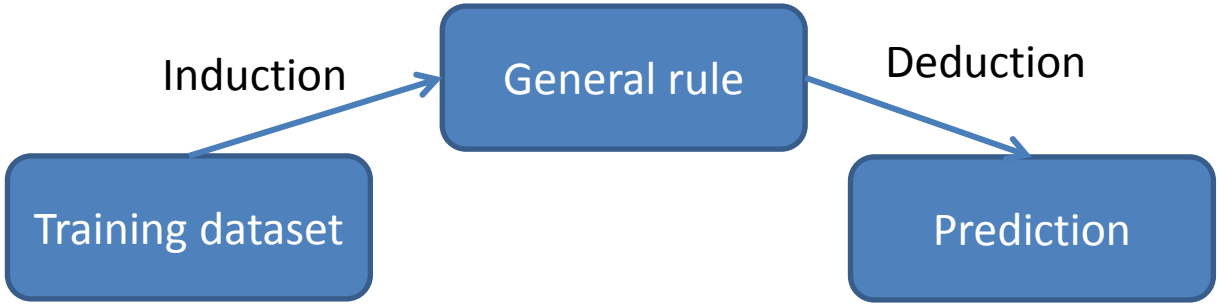
Classification



Regression

Machine learning: classifiers

- Inductive/deductive phases

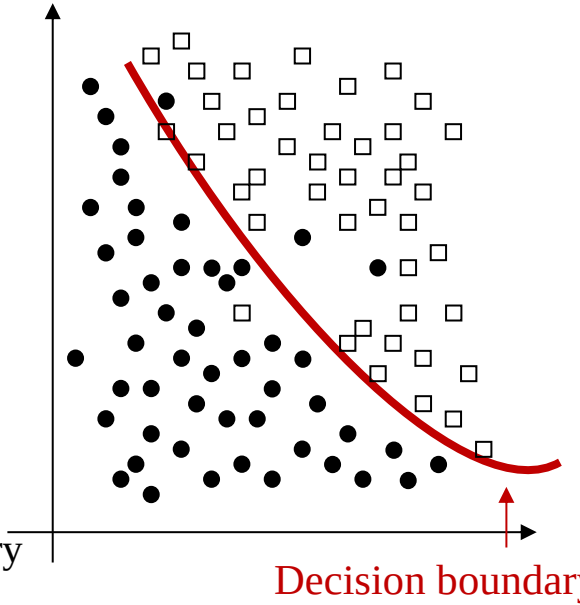


$\{z_i = (\mathbf{x}_i, y_i), i = 1,..., N, \mathbf{x}_i \in \mathbb{R}^m, y_i \in \{L_1, L_2, ...L_k\}\}$

$\mathbf{x}_i$: features of distinctive nature (known)

y_i : labels (known)

In general, the greater N the better



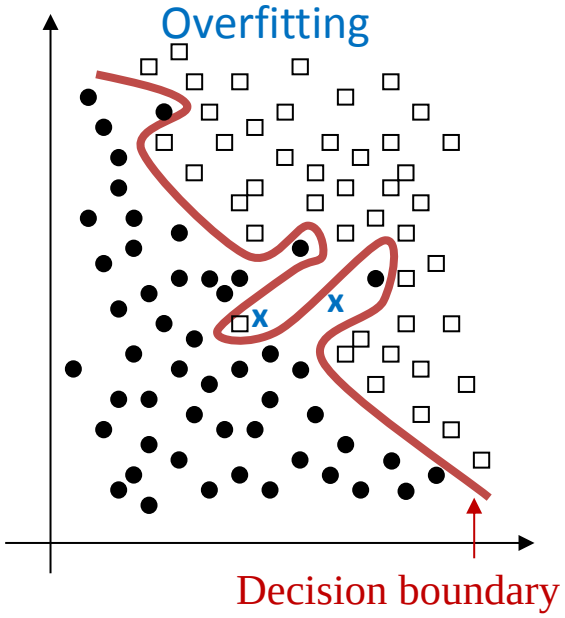
Generalization capability

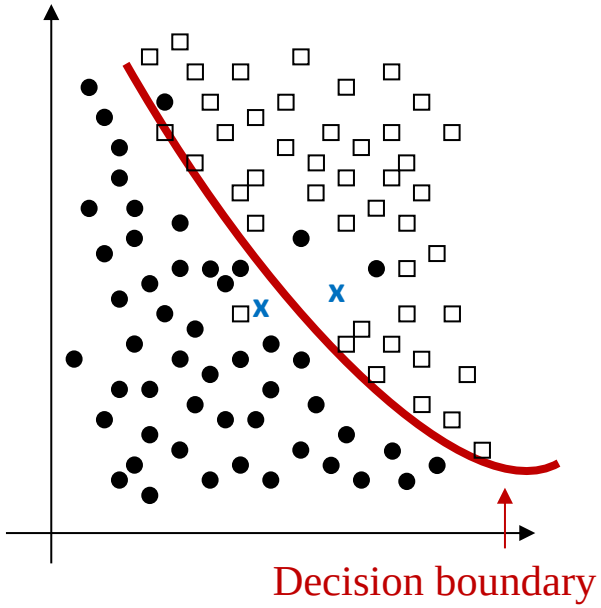
General rule: relationship between features that determines the decision boundary

Example to classify: $\mathbf{x} \in \mathbb{R}^m$ (known), $y \in \{L_1, L_2, ..., L_k\}$ (unknown)

Test dataset: $\{z_i = (\mathbf{x}_i, y_i), i = 1,..., M, \mathbf{x}_i$ and y_i are known $\forall i\}$

Machine learning: generalization capability

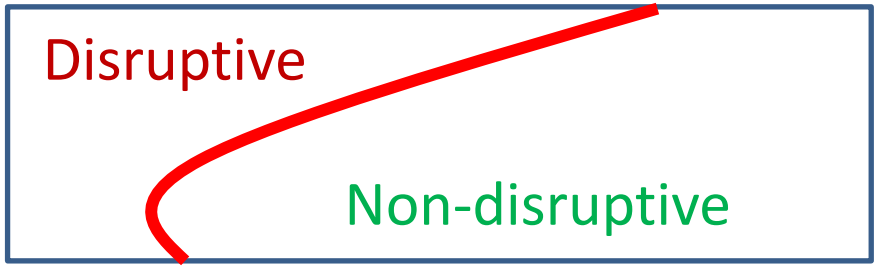




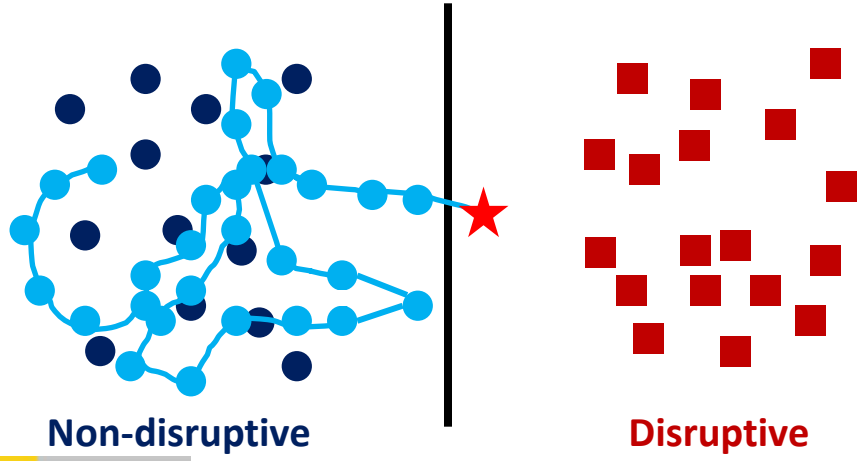
Disruption predictors as classification systems

Disruption predictors based on machine learning techniques

- Disruptive and non-disruptive behaviors have to be distinguished
  - Binary classification problem
    - Training dataset: $\left\{ z_i = (\mathbf{x}_i, y_i), i = 1,..., N, \mathbf{x}_i \in \mathbb{R}^m, y_i \in \{Disruptive, Non-disruptive\} \right\}$
    - Example to classify: $\mathbf{x} \in \mathbb{R}^m$ (known), $y \in \{Disruptive, Non-disruptive\}$ (unknown)



- A multi-dimensional operational space is split into two zones
- A training process determines the separation frontier
- In a running discharge, when a sample is classified in the disruptive zone, an alarm is triggered



Feature vectors for disruption predictors

- Features can be amplitudes extracted from the temporal evolution of the signals

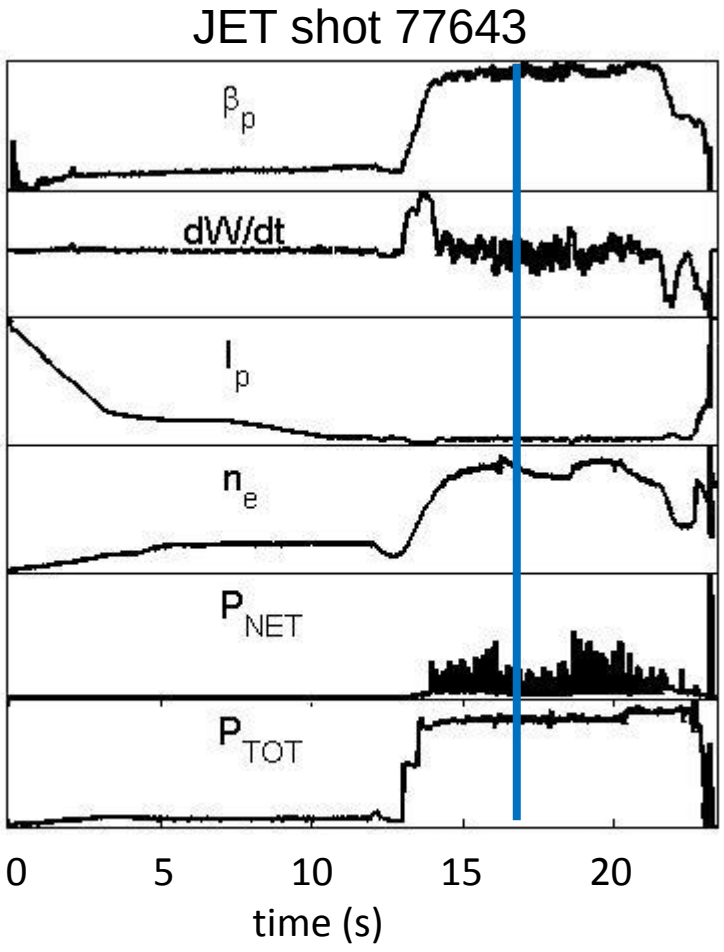
$$\mathbf{x}(t) = \left(\beta_p(t), {}^{dW(t)}\!/_{dt}, I_p(t), n_e(t), P_{NET}(t), P_{TOT}(t) \right)$$

$$\mathbf{x}(t) \in \mathbb{R}^6$$

- The predictor is trained with examples of class *disruptive* and class *non-disruptive*
  - Disruptive examples correspond to times very close to the disruption
- The training process determines a separation frontier between both classes
- The separation frontier is expressed as a relationship among different signals

$$f\left(\beta_p, {}^{dW}\!/_{dt}, I_p, n_e, P_{NET}, P_{TOT} \right)$$

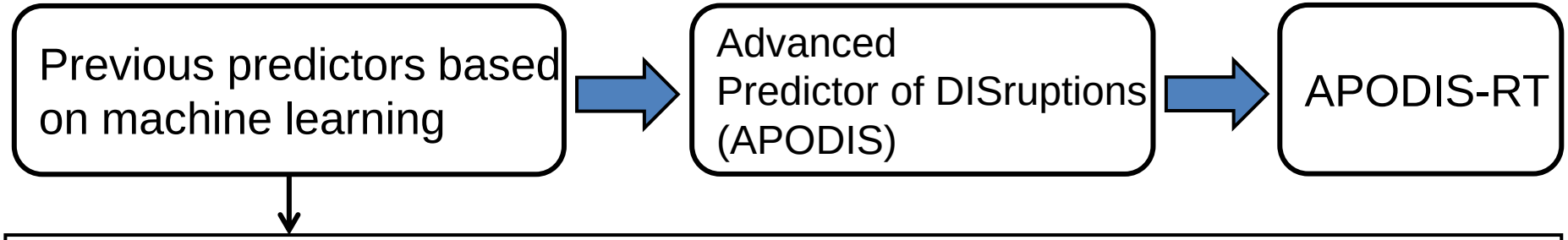
- During a running discharge, feature vectors are generated on a periodic basis and are used as inputs to the predictor
  - The predictor assigns the label '*disruptive*' or '*non-disruptive*' to each feature vector
  - When the label is '*disruptive*' an alarm is triggered



Drawbacks

- Predictors based exclusively on signal amplitudes in the time domain show an ‘ageing’ effect
  - Deterioration of the predictor performance when the device is operated in physics operational regions different from those used for the training
- The combination of amplitudes in particular zones of the physics operational space do not show enough generalization capabilities to be valid in the whole space
  - New type of features are necessary and they have to be common to all disruptions to overcome the dependency of signal amplitudes in the traditional physics operational spaces
- The aim is to configure a more general operational space
  - Disruption precursors produce *anomalous signals*

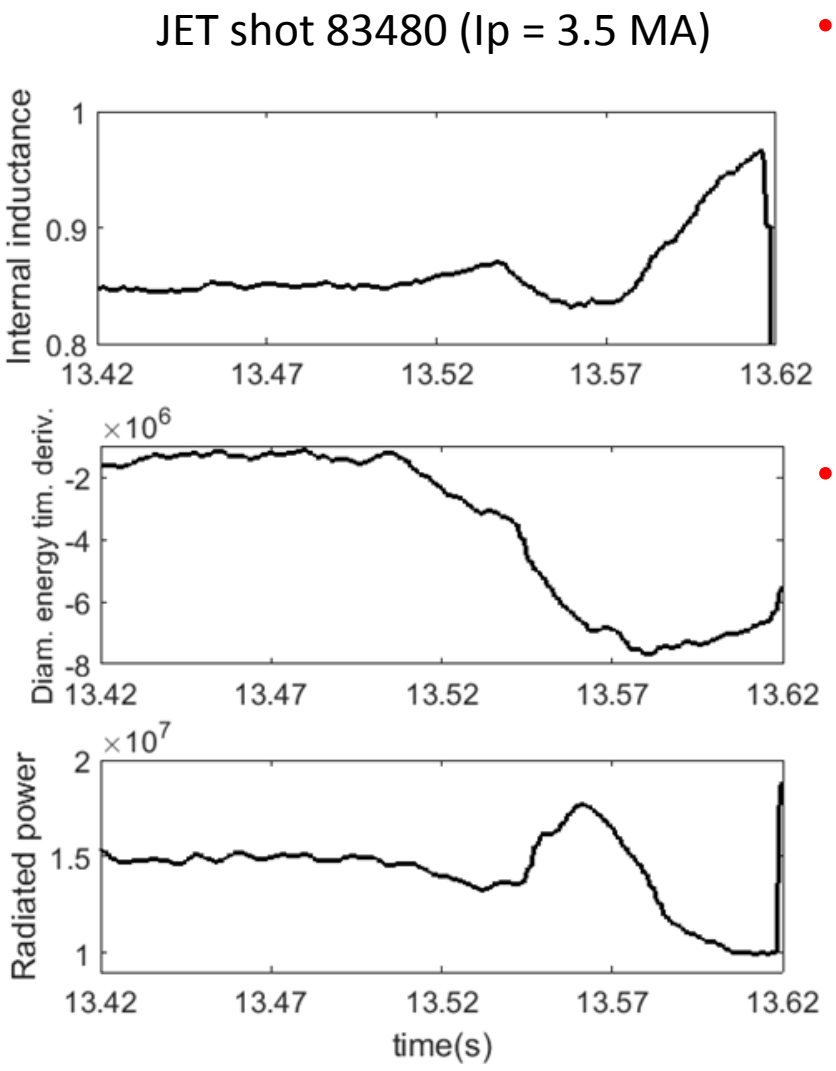
Situation at JET prior to the ILW



- Do not analyse the whole evolution of a shot
  - Models not valid for RT
- Use databases with less than 300 discharges
  - Limited statistical validity
- Do not include all types of disruptions
  - Fastest disruptions are not taken into account
- Cannot extrapolate their results between campaigns
  - Good success rates (~ 90%) with training/test sets of the same campaign
  - Strong degradation appears with discharges belonging to campaigns different to the one of the training dataset
    - success rates ~ 60%

Combination of the temporal domain and the frequency domain of the signals for disruption prediction

Features in the frequency domain



- In this case, disruption precursors are evident in the signals about 100 ms before the disruption
  - The Fourier spectrum of the individual signals change
  - Typically, higher frequency components appear
- If the variation of the Fourier spectrum can be quantified, features in the frequency domain can be candidates to develop disruption predictors
  - The use of the Fourier spectra means to analyze the signals in time windows
    - Minimum number of digitized samples
  - How can be quantified the variation of the Fourier spectrum?
  - The first step is to characterize the Fourier spectrum in each time window
    - Simplest solution: to map each spectrum into a single value

Characterization of the Fourier spectrum of a signal in a time window

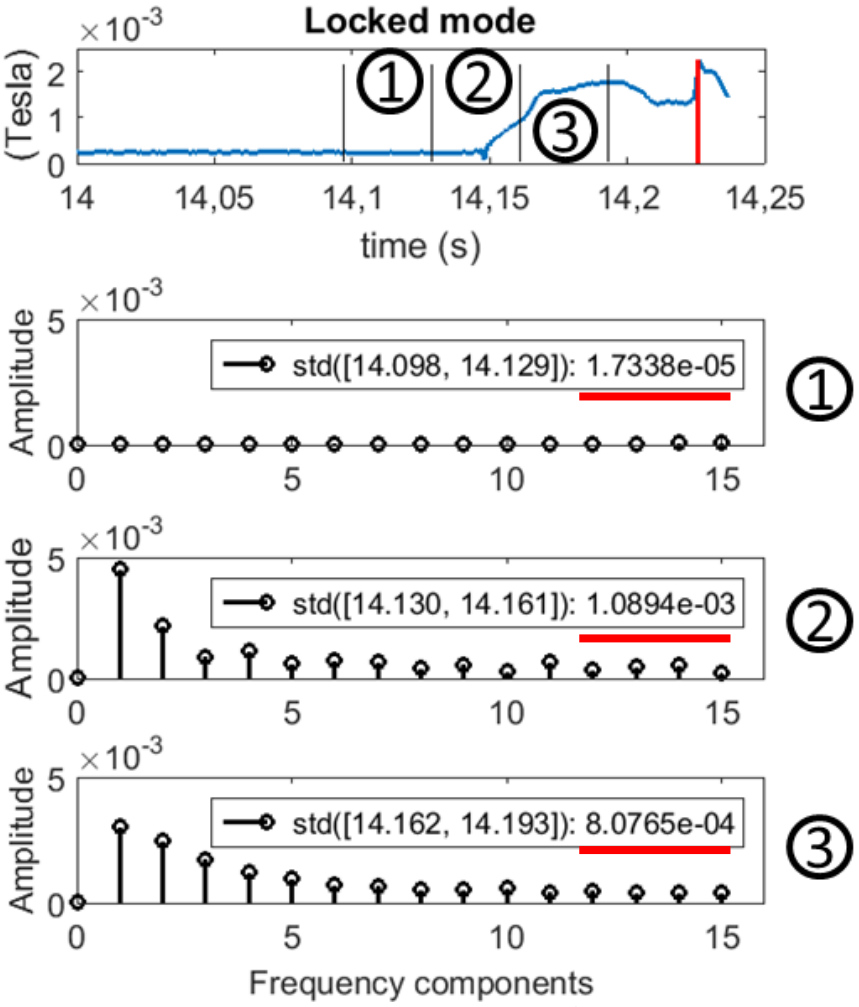
- Standard deviation of the Fourier spectrum (positive frequencies) after removing the DC component

$$\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$$

$$x_i = std\left(\left|fft\left(s_i(t)\right)\right|_{+}^{*}\right), \quad i = 1, \ldots, n$$

- The first disruption predictor ever to take into account the variation of the Fourier spectrum was tested with the JET database

JET shot 84985
Disruption time: 14.226 s
Sampling rate: 1 kS/s
32 samples per time window

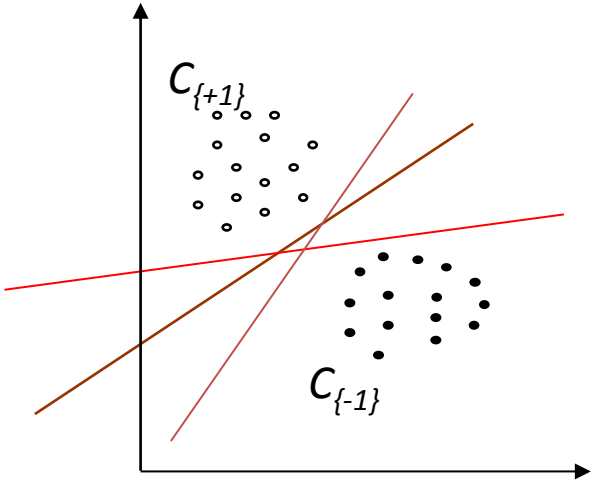


G. A. Rattá, J. Vega, A. Murari et al. "An Advanced Disruption Predictor for JET tested in a simulated Real Time Environment". Nuclear Fusion. 50 (2010) 025005 (10pp)

Introduction to Support Vector Machines

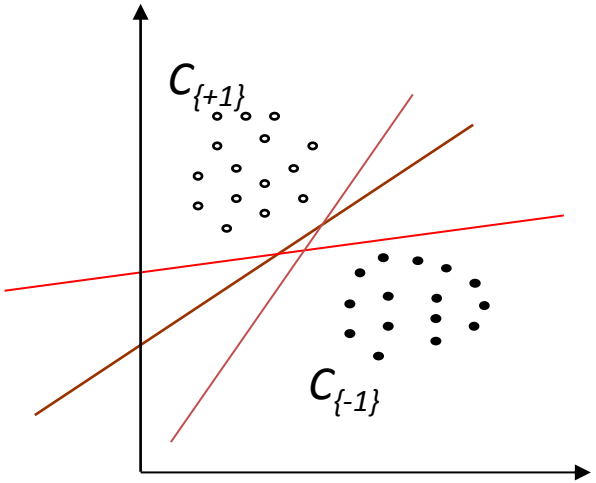
Support Vector Machines (SVM) classifiers

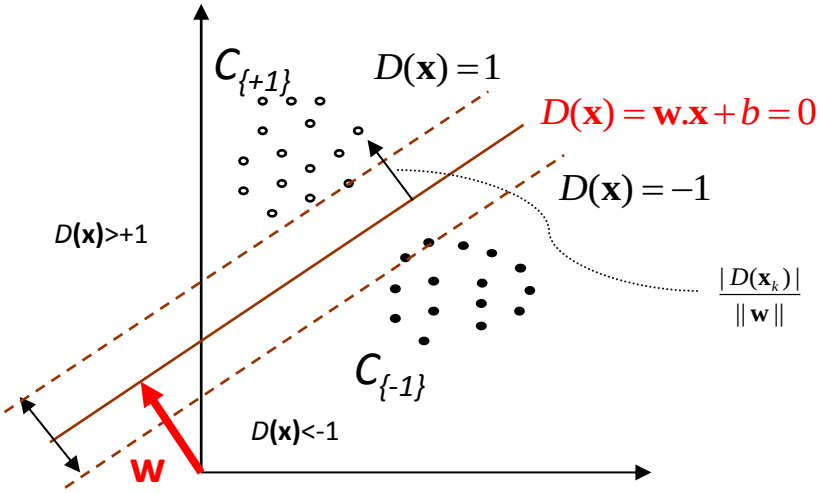
- Binary classifier
- It finds the optimal separating hyper-plane between classes
- Examples: $\left\{ z_i = (\mathbf{x}_i, y_i), i = 1,..., N, \mathbf{x}_i \in \mathbb{R}^m, y_i \in \left\{ C_{\{+1\}}, C_{\{-1\}} \right\} \right\}$



Support Vector Machines (SVM) classifiers

- Binary classifier
- It finds the optimal separating hyper-plane between classes
- Examples: $\left\{ z_i = (\mathbf{x}_i, y_i), i = 1,\ldots,N, \mathbf{x}_i \in \mathbb{R}^m, y_i \in \left\{ C_{\{+1\}}, C_{\{-1\}} \right\} \right\}$





Maximum margin: 2τ

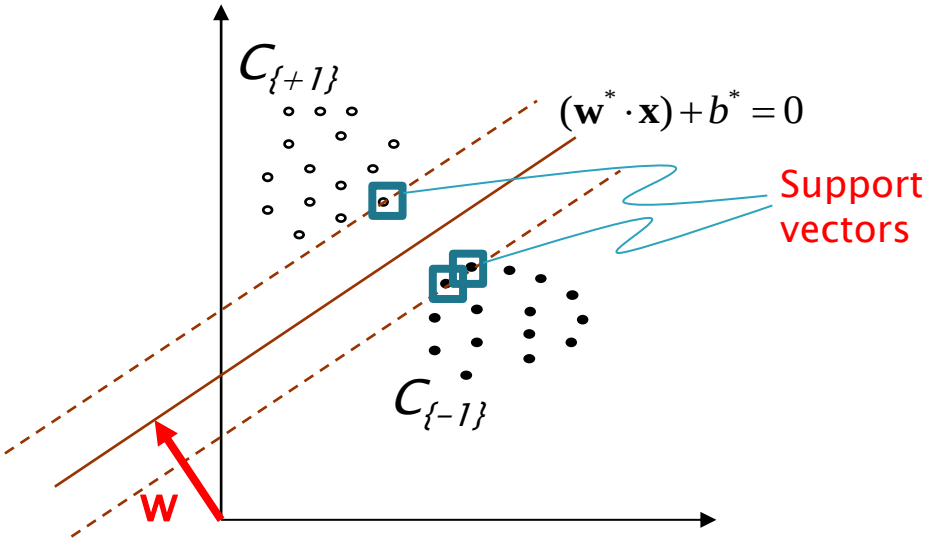
$$\frac{y_k D(\mathbf{x}_k)}{\|\mathbf{w}\|} \geq \tau, \quad y_k \in \{+1, -1\}, \quad k = 1, \cdots, N$$

- To find the optimal hyper-plane it is necessary to determine the vector **w** that maximizes the margin τ
- There are infinite solutions due to the presence of a scale factor. To avoid this: $\tau \, ||\, \mathbf{w}\, || = 1$
- Therefore, to maximize the margin is equivalent to minimize $||\, \mathbf{w}\, ||$
- **Optimization problem:**

$$\min_{\mathbf{w}, w_0} \; J(\mathbf{w}) = \|\mathbf{w}\|^2, \quad \text{subject to} \quad y_k \left[(\mathbf{w}.\mathbf{x}_i) + w_0 \right] \geq 1$$

Support Vector Machines (SVM) classifiers

$$\left\{ z_i = (\mathbf{x}_i, y_i), i = 1,\ldots,N, \mathbf{x}_i \in \mathbb{R}^m, y_i \in \left\{ C_{\{+1\}}, C_{\{-1\}} \right\} \right\}$$



- Solution: $\mathbf{w}^* = \sum_{i=1}^{N} \alpha_i^* y_i \mathbf{x}_i$

  α_i are the Lagrange multipliers

- Samples associated to $\alpha_i \neq 0$ are called "support vectors"

$$\mathbf{w} = \sum_{\text{support vectors}} \alpha_i^* y_i \mathbf{x}_i$$

The rest of training samples are irrelevant to classify new samples

- The constant b is obtained from any condition (Karush-Kuhn-Tucker)

$$\alpha_i \left[y_i \left((\mathbf{w} \cdot \mathbf{x}_i) + b \right) - 1 \right] = 0, \quad i = 1,\ldots,N$$

$D(\mathbf{x}) = \mathbf{w}^* \cdot \mathbf{x} + b^*$ is the distance (with sign) from **X** to the separating hyper-plane

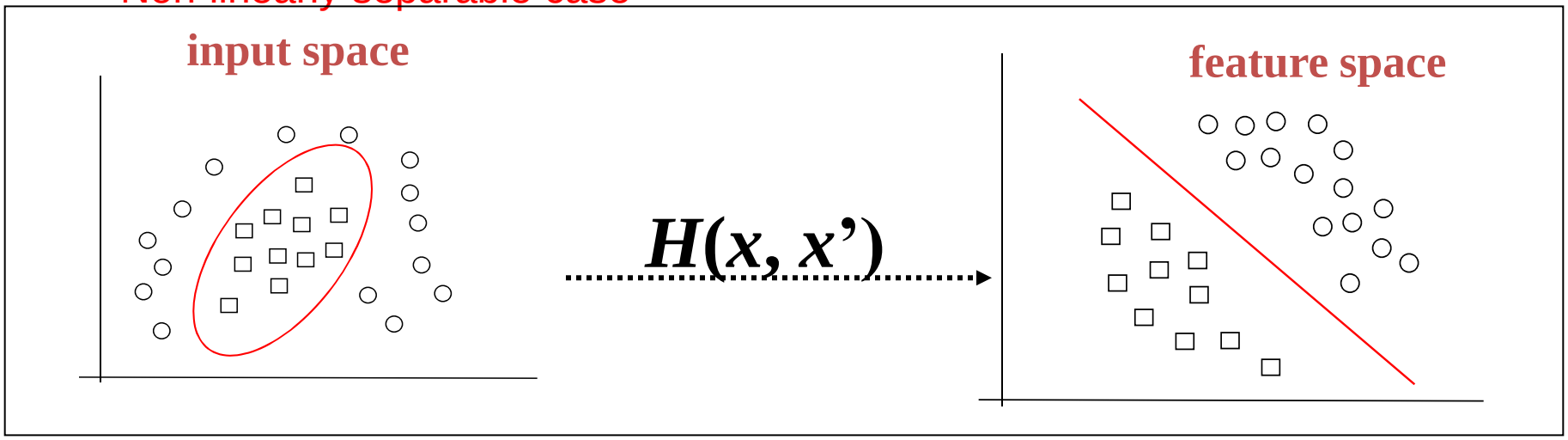
Given **x** to classify

$$\text{if } sign\left(\sum_{\text{vectores soporte}} \alpha_i^* y_i (\mathbf{x} \cdot \mathbf{x}_i) + b^* \right) \geq 0, \quad \mathbf{x} \in C_{\{+1\}}. \quad \text{Otherwise } \mathbf{x} \in C_{\{-1\}}$$

V. Cherkassky, F. Mulier. *Learning from data*. 2nd edition. Wiley-Interscience.

Support Vector Machines (SVM) classifiers

- Non-linearly separable case



- Kernels
  - Linear: $H(\mathbf{x},\mathbf{x}')=(\mathbf{x}.\mathbf{x}')$
  - Polynomials of degree q: $H(\mathbf{x},\mathbf{x}')=[(\mathbf{x}.\mathbf{x}')+1]^q$
  - Radial basis functions: $\longrightarrow H(\mathbf{x},\mathbf{x}')=\exp\left\{-\frac{|\mathbf{x}-\mathbf{x}'|^2}{\sigma^2}\right\}$
  - Neural network: $H(\mathbf{x},\mathbf{x}')=\tanh(2(\mathbf{x}.\mathbf{x}')+1)$

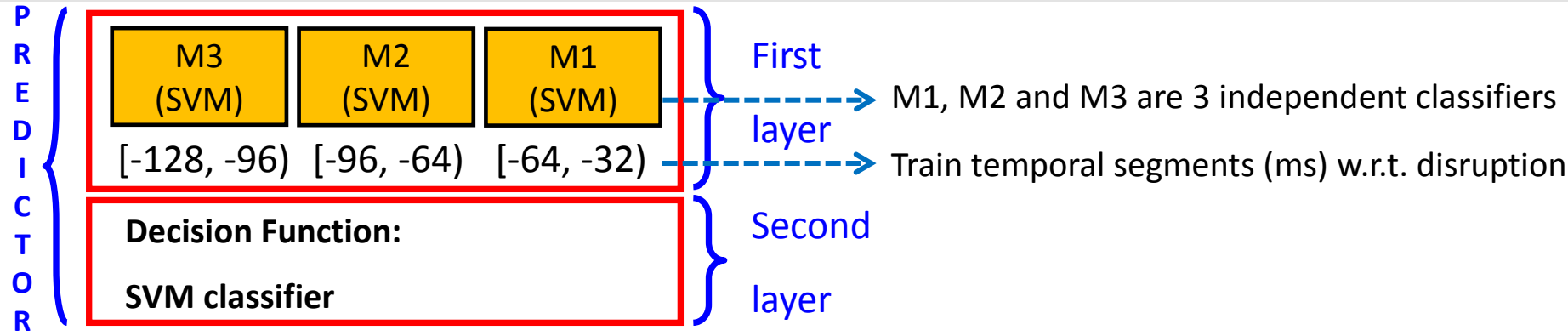
Given $\mathbf{x}$ to classify

$$if\ sign\left(\sum_{vectores\ soporte}\alpha_i^* y_i H(\mathbf{x},\mathbf{x}_i)+b^*\right)\geq 0,\quad \mathbf{x}\in C_{\{+1\}}.\quad \text{Otherwise } \mathbf{x}\in C_{\{-1\}}$$

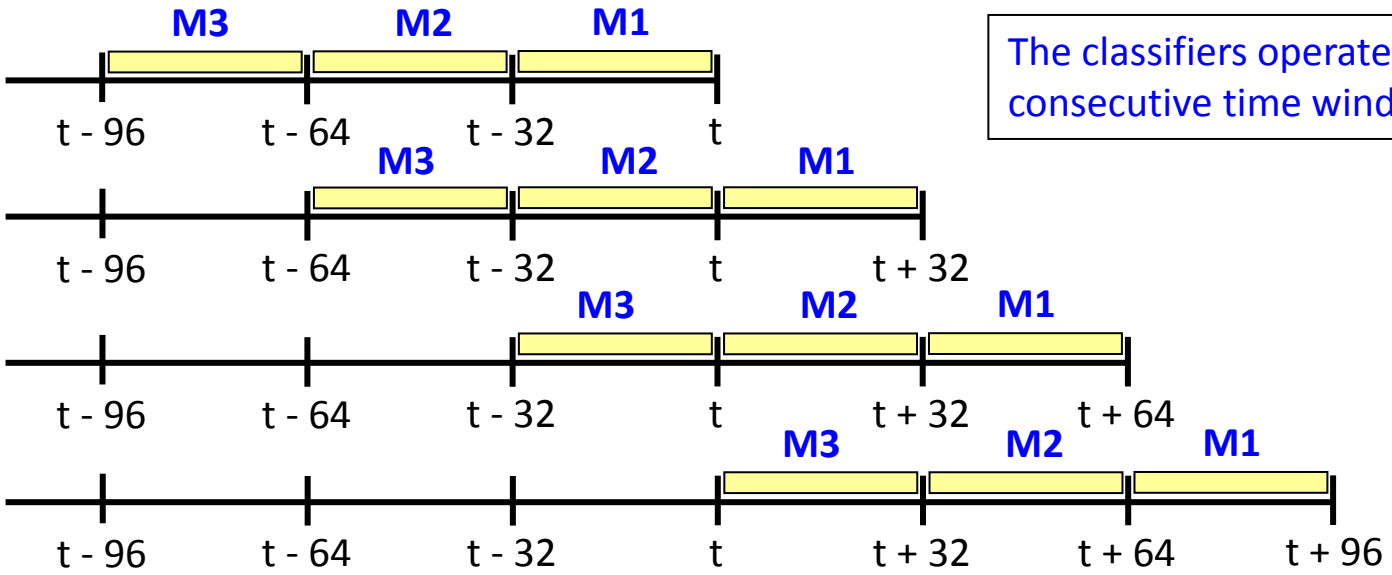
V. Cherkassky, F. Mulier. *Learning from data*. 2nd edition. Wiley-Interscience.

APODIS disruption predictor

Advanced Predictor Of DISruptions: APODIS architecture



- As a discharge is in execution, *the most recent 32 ms temporal segments* are classified as disruptive or non-disruptive



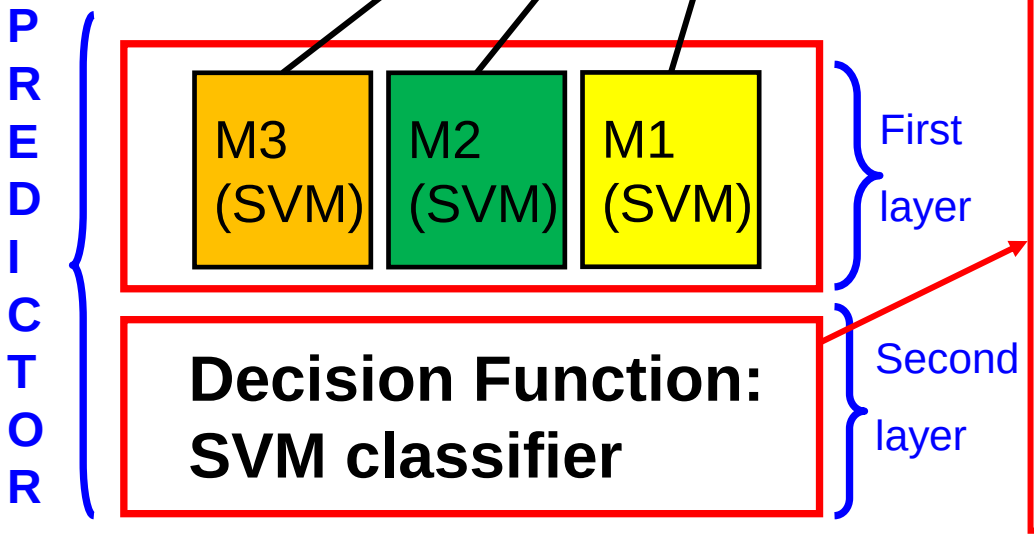
The classifiers operate in parallel on consecutive time windows

- The three classifiers may disagree about the discharge behaviour ⇨ 2nd layer

APODIS details (1/2)

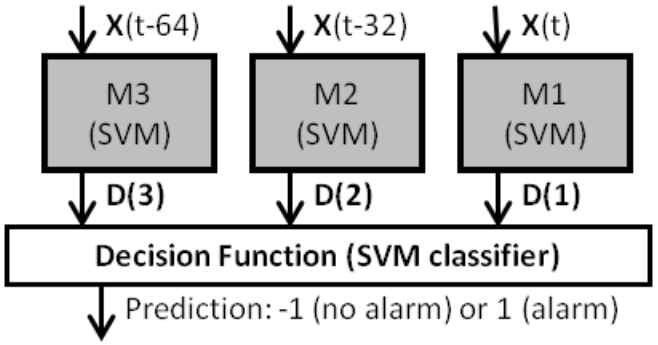
Two-layer classifier to identify **early enough** precursors of disruptions

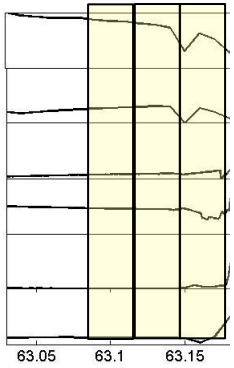
- Training phase: determining a **model** (separating hyper-plane) between the two behaviours (non-disruptive and disruptive)
  - Input: Feature vectors of 13 components (supervised manner)
  - Output: hyper-plane equation
- Real time prediction: uses the trained model
  - Input: all feature vector of 13 components along the discharge
  - Output: D/ND character of the time slice and the distance to the separating hyper-plane (the larger absolute value the more certain is the prediction)

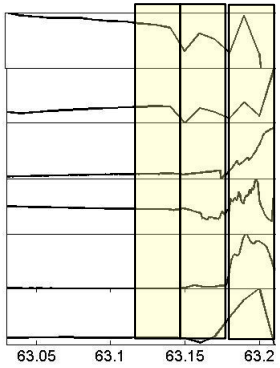


- Training phase: determining a **model** (separating hyper-plane) between the two behaviours (non-disruptive and disruptive)
  - Input: outputs of the first layer (D/ND character and distances to the respective hyper-planes)
  - Output: hyper-plane equation
- Real time prediction: uses the trained model
  - Input: outputs of the first layer
  - Output: decision to trigger an alarm or not

APODIS details (2/2)







$$D(M) = K(M) + \sum_{i=1}^{N(M)} \alpha_i(M) \cdot \exp\left\{-\left|\mathbf{v}_i(M) - \mathbf{x}\right|^2\right\}$$

$M = 1, 2, 3$

$D(M)$: distance (with sign) corresponding to model M

$K(M)$: bias of model M

$N(M)$: number of support vectors for model M

$\alpha_i(M)$: Lagrange multipliers corresponding to the support vectors of model M

$\mathbf{v}_i(M)$: support vectors of model M

$\mathbf{x}$: input feature vector

$$F_P = sign\left\{B_0 + B_1 \cdot D(1) + B_2 \cdot D(2) + B_3 \cdot D(3)\right\}$$

B_j, $j = 0, 1, 2, 3$ are the hyper-planes coefficients obtained in the training of the second layer classifier

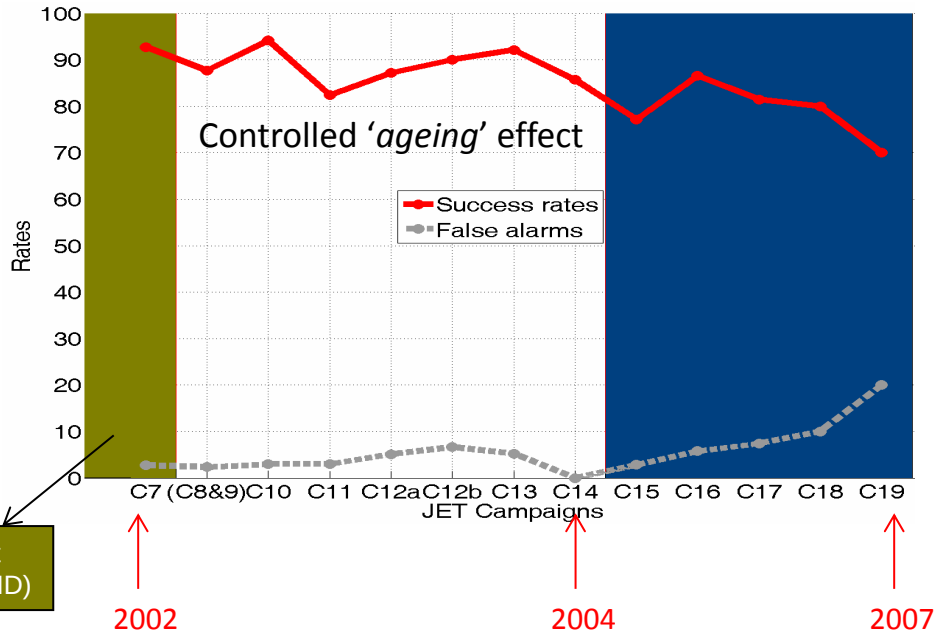
APODIS first results

#	Signal name	Units
1	Plasma current	A
2	Poloidal beta	
3	Poloidal beta time derivative	s^{-1}
4	Locked mode amplitude	T
5	Safety factor at 95% of minor radius	
6	Safety factor at 95% of minor radius time derivative	s^{-1}
7	Total input power	W
8	Plasma internal inductance	
9	Plasma internal inductance time derivative	s^{-1}
10	Plasma vertical centroid position	m
11	Plasma density	m^{-3}
12	Stored diamagnetic energy time derivative	W
13	Net power (total input power minus total radiated power)	W

- The first APODIS version was trained/tested with 13 signals
- Feature vectors

$$\mathbf{x} = \left(x_1, ..., x_{13} \right) \in \mathbb{R}^{13}$$

$$x_i = std\left(\left| fft\left(s_i\left(t\right)\right)\right|_+^* \right)$$



Unique training dataset
438 shots: 263 (D)+175 (ND)

APODIS in the JET real-time network

- Only real-time quantities are used
- Only seven signals
- Two kinds of features per signal
  - Sampling rate: 1 kS/s
  - Time windows 32 ms long

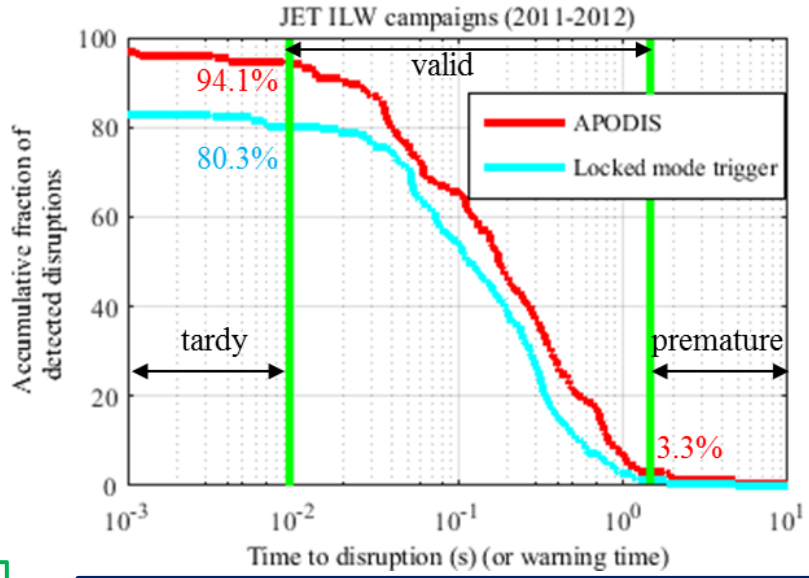
$$\mathbf{x} = (x_1, x_2, ..., x_{14}) \in \mathbb{R}^{14}$$

$$x_{2k-1} = mean(s_i(t)), \quad x_{2k} = std\left(\left|fft(s_i(t))\right|_+^*\right)$$

$$k = 1, 2, ..., 7$$

- Training dataset: the largest number of discharges ever: 8169 (2006-2009, **C-wall**)
  - 7648 non-disruptive
  - 521 disruptive

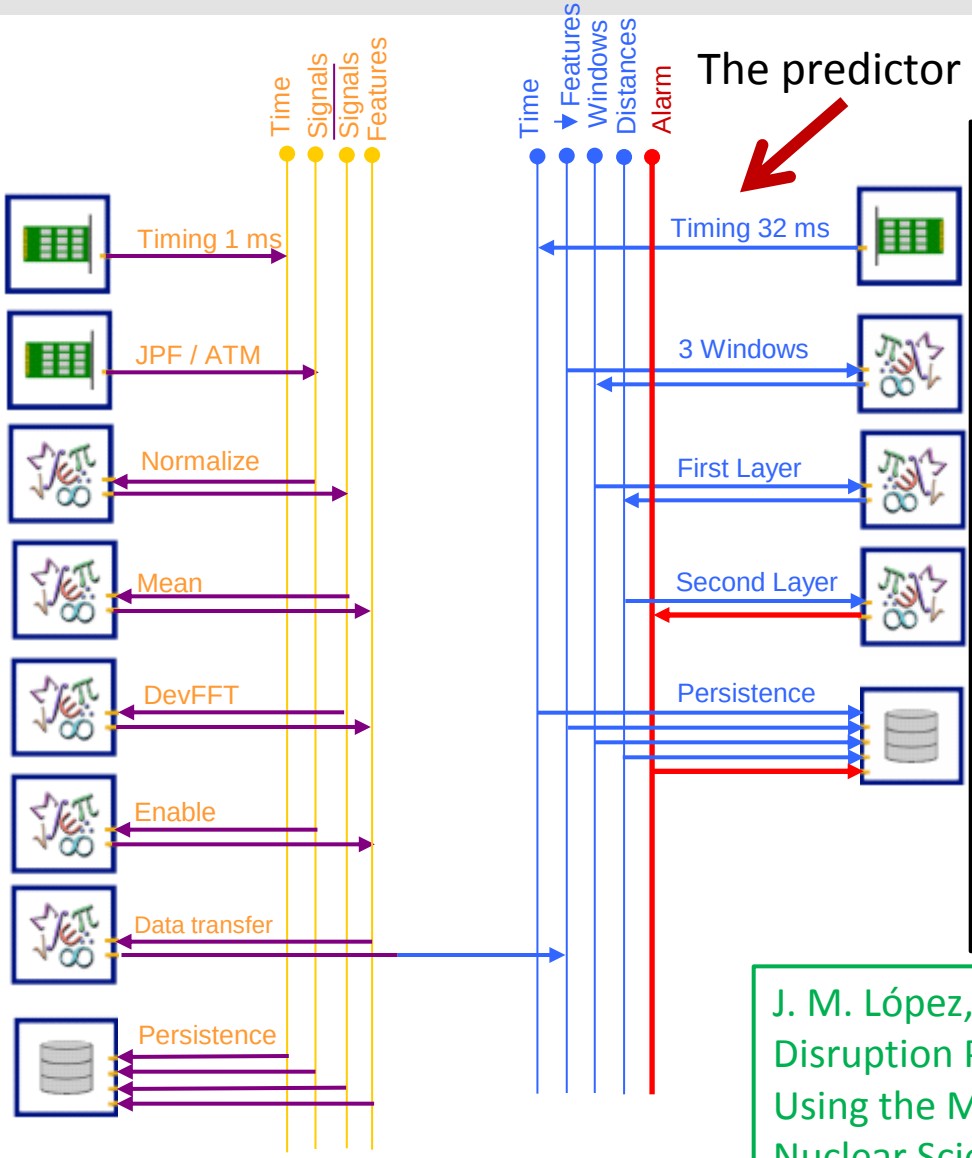
#	Signal name	Units
1	Plasma current	A
2	Locked mode amplitude	T
3	Total input power	W
4	Plasma internal inductance	
5	Plasma density	m^{-3}
6	Stored diamagnetic energy time derivative	W
7	Radiated power	W



- No retraining
- 1237 shots (1036 safe/201 disruptive)
- False alarms: 1%
- Average warning time: 426 ms

J. Vega, S. Dormido-Canto, J. M. López et al. "Results of the JET real-time disruption predictor in the ITER-like wall campaigns", Fus. Eng. Des. 88 (2013) 1228-1231

APODIS implementation in the JET RT MARTe framework



The predictor computations are carried out in about 300 μs

- Implementation under the Multithreaded Application Real-Time executor (MARTe) on a six-core x86 architecture
- 2 applications threads
  - One collects the samples from the input sources and forms the feature vectors
  - The other
    - receives the data from the first thread
    - organizes them to fit in the three windows architecture
    - evaluates the 4 SVM classifiers

J. M. López, J. Vega, D. Alves et al. "Implementation of the Disruption Predictor APODIS in JET's Real-Time Network Using the MARTe Framework". IEEE Transactions on Nuclear Science. 61, 2 (2014), 741-744

Predictors from scratch

ITER requirements

- APODIS training/test datasets: 2006-2009
  - 7648 non-disruptive + 521 unintentional disruptions
- ITER cannot wait for hundreds of disruptions to have a reliable predictor
- ITER needs adaptive disruption predictors '*from scratch*'
  - '*from scratch*' means 'absolute lack of previous information'
    - New fusion devices (ITER, DEMO)
    - Existing devices with significant changes (JET with the ILW)
  - Incorporation of new knowledge is accomplished according to the chronological order of the discharges

Prediction from scratch

- The objective of predictors from scratch is to avoid the dependency on large databases from past discharges to be able to recognize disruption precursors

- Predictors from scratch start to make predictions after a first disruption
  - Typically more than one non-disruptive discharge will be available

- New predictors are created in an adaptive way after each missed alarm

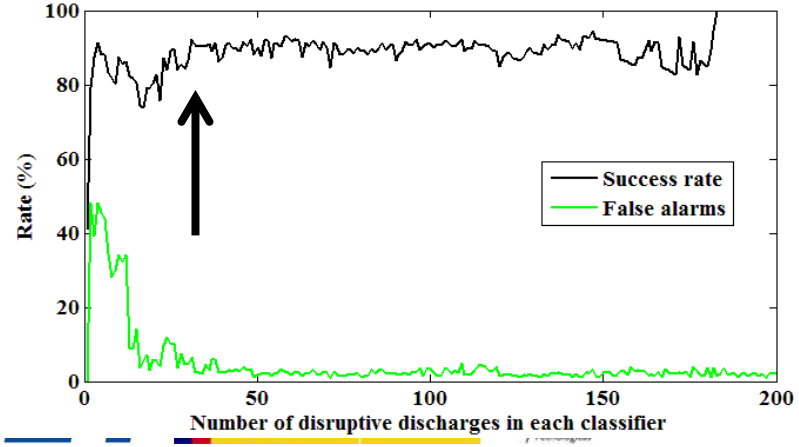
First predictor from scratch with JET data

- **Objective**: determining the number of disruptions that are needed to have a reliable predictor from scratch with APODIS
- Only real-time quantities are used
- 12 signals
- Two kinds of features per signal
  - Sampling rate: 1 kS/s
  - Time windows 32 ms long
- 1237 discharges (2011-2012)
  - 1036 non-disruptive/201 disruptions

$$\mathbf{x} = (x_1, x_2, ..., x_{24}) \in \mathbb{R}^{24}$$

$$x_{2k-1} = mean(s_i(t)), \quad x_{2k} = std\left(|fft(s_i(t))|_+^*\right)$$

$$k = 1, 2, ..., 12$$



#	Signal name	Units
1	Plasma current	A
2	Poloidal beta	
3	Poloidal beta time derivative	s^{-1}
4	Locked mode amplitude	T
5	Total input power	W
6	Plasma internal inductance	
7	Plasma internal inductance time derivative	s^{-1}
8	Plasma vertical centroid position	m
9	Plasma density	m^{-3}
10	Stored diamagnetic energy time derivative	W
11	Radiated power	W
12	Plasma vertical centroid position time derivative	m/s

S. Dormido-Canto, J. Vega, J. M. Ramírez et al. "Development of an efficient real-time disruption predictor from scratch on JET and implications for ITER". Nuclear Fusion 53 (2013) 113001 (8pp).

High learning rate predictor from scratch: requirements

- High learning rate
  - The first model will have only 1 disruptive example and 1 non-disruptive example
- Learning processes with really low number of examples
- Fast training processes (from a computational point of view)
  - Inter-shot trainings when necessary
- Simplicity
  - 1 single classifier
  - Very condensed information in the feature space (compatible with the best possible generalization capability to distinguish between behaviours)
  - Reduced number of features
- Reliable predictions
  - The low number of samples is an issue
  - Each individual prediction ***must be*** qualified not only with a probability but with an error bar

Introduction to Venn predictors

Venn prediction

- Training examples: $(\mathbf{x}_i, y_i)$, i = 1, ..., n-1
  - $\mathbf{x}_i \in \mathbb{R}^m$ (known): feature of distinctive nature
  - $y_i \in \{L_1, L_2, ..., L_C\}$ (known)
- Example to classify: $(\mathbf{x}, y)$
  - $\mathbf{x} \in \mathbb{R}^m$ (known)
  - $y \in \{L_1, L_2, ..., L_C\}$ (**unknown**)

The Venn prediction framework assigns ***each one*** of the possible classifications L_j, j = 1, ..., C to **x** and **x** is placed in the training dataset. Therefore, C different training datasets are considered that have to be divided into a number of categories based on a taxonomy

$(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{n-1}, y_{n-1}), (\mathbf{x}, L_1)$

$(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{n-1}, y_{n-1}), (\mathbf{x}, L_2)$

.....

$(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{n-1}, y_{n-1}), (\mathbf{x}, L_C)$

The samples are partitioned into T categories $\{\tau_1, ..., \tau_T\}$ using a specific taxonomy (classification criterion with T classes, T can be ≠ C)

$\tau_1=\{(\mathbf{x}_3, y_3), ..., (\mathbf{x}_{20}, y_{20})\}$: N_1 samples

.....

$\tau_j=\{(\mathbf{x}_5, y_5), ..., (\mathbf{x}, L_1), ..., (\mathbf{x}_{40}, y_{40})\}$: N_j samples

.....

$\tau_T=\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_7, y_7)\}$: N_T samples

$$\sum_{j=1}^{T} N_j = n$$

$$\left(p^{L_2}(L_1), ..., p^{L_2}(L_C)\right)$$

$$\left(p^{L_C}(L_1), ..., p^{L_C}(L_C)\right)$$

After determining the category τ inside which **x** is located, the probabilities of each label within the category τ are computed. Therefore, a row vector with C components (one per label) is obtained

$$\left(p^{L_1}(L_1), ..., p^{L_1}(L_C)\right) \quad \text{where} \quad p^{L_1}(L_k) = \frac{\left|\left\{(\mathbf{x}^*, y^*) \in \tau : y^* = L_k\right\}\right|}{|\tau_j|}, k = 1, ..., C$$

Venn prediction

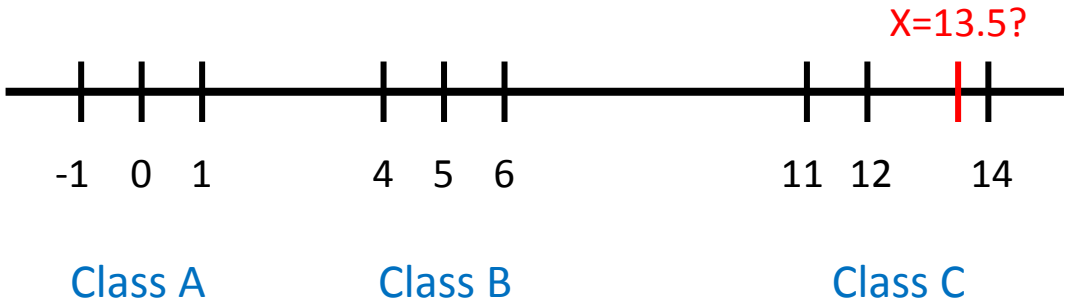
A square matrix is formed to make a prediction

$$P_C = \left(\begin{array}{cc:c} p^{L_1}(L_1) & p^{L_1}(L_2) & p^{L_1}(L_C) \\ p^{L_2}(L_1) & p^{L_2}(L_2) & p^{L_2}(L_C) \\ \hdashline p^{L_C}(L_1) & p^{L_C}(L_2) & p^{L_C}(L_C) \end{array} \right)$$

- There are several criteria to determine the '*winner column*'
  - Highest *quality*
  - Max(mean(columns))
  - Max(median(columns))
- The '*winner column*' establishes the label of the example
- The minimum and maximum probability in this column define the probability interval

- V. Vovk et al. *Algorithmic learning in a random world*. Springer (2005)
- C. Zhou et al. EANN/AIAI 2011, Part II, IFIP AICT 364, pp. 483-490. 2011
- A. Lambrou et al. AIAI 2012 Workshops, IFIP AICT 382, pp. 182-191. Springer, 2012
- I. Nouretdinov et al. AIAI 2012 Workshops, IFIP AICT 382, pp. 224-233. Springer, 2012
- H. Papadopoulos. Neurocomputing 107 (2013) 59-68

Example of a Venn predictor with the nearest centroid taxonomy



What is the class and the probability interval of the prediction of element x=13.5 by using a Venn predictor with the nearest centroid taxonomy?

A Venn taxonomy based on the nearest centroid is used. This means that the category of an example is equal to the label of its nearest centroid. Therefore, there are as many categories as labels: $Y = \{A, B, C\}$

Categories: $\{\tau_A, \tau_B, \tau_C\}$

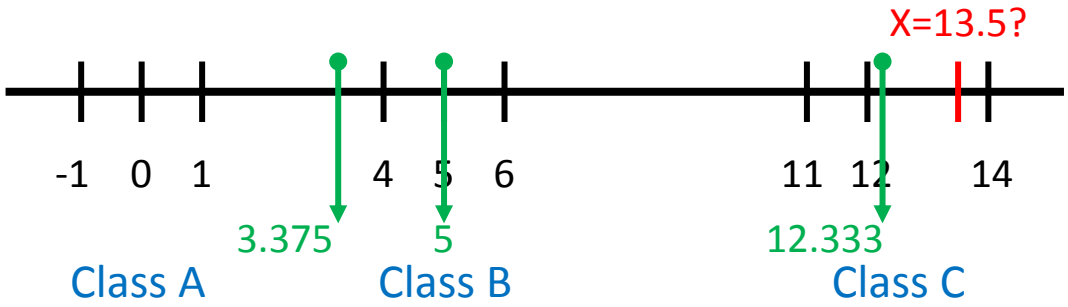
$$P_C = \begin{pmatrix} p^A(A) & p^A(B) & p^A(C) \\ p^B(A) & p^B(B) & p^B(C) \\ p^C(A) & p^C(B) & p^C(C) \end{pmatrix}$$

← 1st step: Let's assume (13.5, **A**)

← 2nd step: Let's assume (13.5, **B**)

← 3rd step: Let's assume (13.5, **C**)

1st step: (13.5, A)



$$Centroid(A) = \frac{-1+0+1+13.5}{4} = 3.375 \Rightarrow (3.375, A)$$

$$Centroid(B) = \frac{4+5+6}{3} = 5 \Rightarrow (5, B)$$

$$Centroid(C) = \frac{11+12+14}{3} = 12.3333 \Rightarrow (12.3333, C)$$

The category of an example is equal to the label of its nearest centroid

Element	Nearest centroid and label	Category
(-1, A)	(3.375, A)	τ_A
(0, A)	(3.375, A)	τ_A
(1, A)	(3.375, A)	τ_A
(4, B)	(3.375, A)	τ_A
(5, B)	(5, B)	τ_B
(6, B)	(5, B)	τ_B
(11, C)	(12.333, C)	τ_C
(12, C)	(12.333, C)	τ_C
(14, C)	(12.333, C)	τ_C
(13.5, A)	(12.333, C)	τ_C

$$\tau_A = \{(-1, A), (0, A), (1, A), (4, B)\}$$

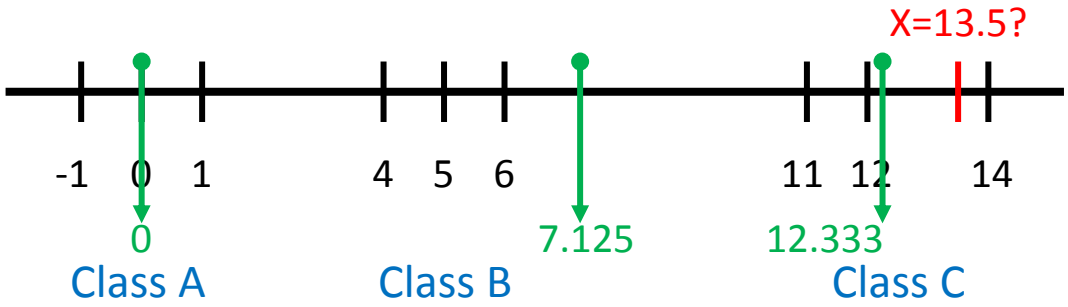
$$\tau_B = \{(5, B), (6, B)\}$$

$$\tau_C = \{(11, C), (12, C), (14, C)(13.5, A)\}$$

The element (13.5, A) is in category τ_C and, therefore, the first row of P_C is computed with the data corresponding to category τ_C

$$P^A(A) = 1/4, \quad P^A(B) = 0, \quad P^A(C) = 3/4$$

2nd step: (13.5, B)



$$Centroid(A) = \frac{-1+0+1}{3} = 0 \Rightarrow (0, A)$$

$$Centroid(B) = \frac{4+5+6+13.5}{4} = 7.125 \Rightarrow (7.125, B)$$

$$Centroid(C) = \frac{11+12+14}{3} = 12.3333 \Rightarrow (12.3333, C)$$

The category of an example is equal to the label of its nearest centroid

Element	Nearest centroid and label	Category
(-1, A)	(0, A)	τ_A
(0, A)	(0, A)	τ_A
(1, A)	(0, A)	τ_A
(4, B)	(7.125, B)	τ_B
(5, B)	(7.125, B)	τ_B
(6, B)	(7.125, B)	τ_B
(11, C)	(12.333, C)	τ_C
(12, C)	(12.333, C)	τ_C
(14, C)	(12.333, C)	τ_C
(13.5, B)	(12.333, C)	τ_C

$$\tau_A = \{(-1, A), (0, A), (1, A)\}$$

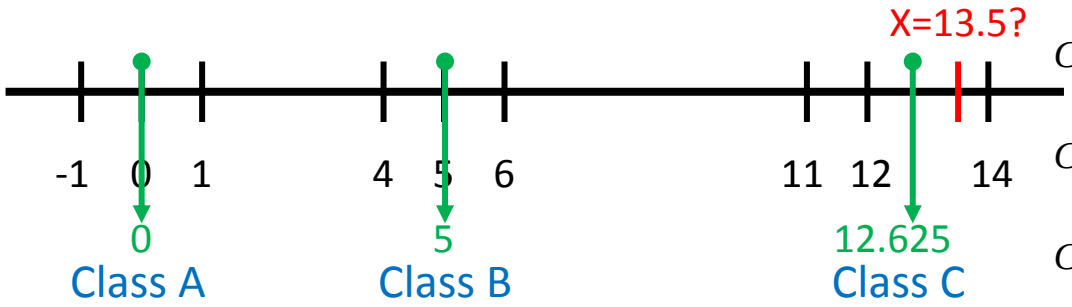
$$\tau_B = \{(4, B), (5, B), (6, B)\}$$

$$\tau_C = \{(11, C), (12, C), (14, C)(13.5, B)\}$$

The element (13.5, B) is in category τ_C and, therefore, the second row of P_C is computed with the data corresponding to category τ_C

$$P^B(A) = 0, \quad P^B(B) = 1/4, \quad P^B(C) = 3/4$$

3rd step: (13.5, C)



$$Centroid(A) = \frac{-1+0+1}{3} = 0 \Rightarrow (0, A)$$

$$Centroid(B) = \frac{4+5+6}{3} = 5 \Rightarrow (5, B)$$

$$Centroid(C) = \frac{11+12+14+13.5}{4} = 12.625 \Rightarrow (12.625, C)$$

The category of an example is equal to the label of its nearest centroid

Element	Nearest centroid and label	Category
(-1, A)	(0, A)	τ_A
(0, A)	(0, A)	τ_A
(1, A)	(0, A)	τ_A
(4, B)	(5, B)	τ_B
(5, B)	(5, B)	τ_B
(6, B)	(5, B)	τ_B
(11, C)	(12.625, C)	τ_C
(12, C)	(12.625, C)	τ_C
(14, C)	(12.625, C)	τ_C
(13.5, C)	(12.625, C)	τ_C

$$\tau_A = \{(-1, A), (0, A), (1, A)\}$$

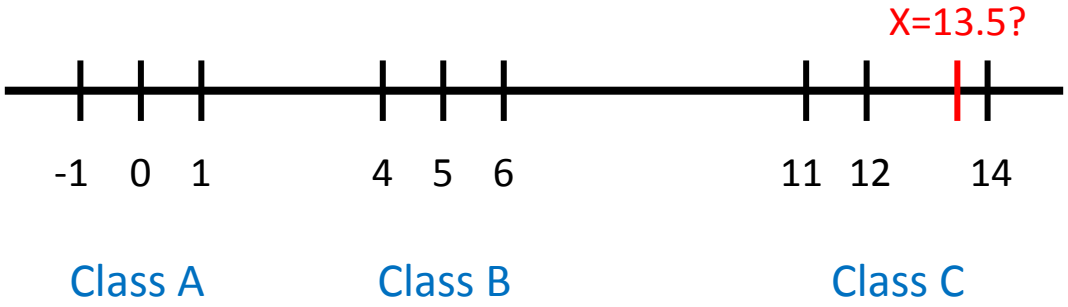
$$\tau_B = \{(4, B), (5, B), (6, B)\}$$

$$\tau_C = \{(11, C), (12, C), (14, C)(13.5, C)\}$$

The element (13.5, C) is in category τ_C and, therefore, the third row of P_C is computed with the data corresponding to category τ_C

$$P^C(A) = 0, \quad P^C(B) = 0, \quad P^C(C) = 1$$

Label prediction and probability interval



- To determine the '*winner column*', the Max(mean(columns)) criterion is chosen

$$P_C = \begin{pmatrix} p^A(A) & p^A(B) & p^A(C) \\ p^B(A) & p^B(B) & p^B(C) \\ p^C(A) & p^C(B) & p^C(C) \end{pmatrix} = \begin{pmatrix} 1/4 & 0 & 3/4 \\ 0 & 1/4 & 3/4 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\overline{p(A)} = \frac{1/4+0+0}{3} = \frac{1}{12} = 0.0833 \qquad \overline{p(B)} = \frac{0+1/4+0}{3} = \frac{1}{12} = 0.0833 \qquad \overline{p(C)} = \frac{3/4+3/4+1}{3} = \frac{10}{12} = 0.8333$$

The Venn outputs the prediction $\hat{y}_n = \{C\}$

The probability interval for this prediction is $\left[\frac{3}{4}, 1\right]$

Second predictor from scratch with JET data

- This predictor uses the **Venn prediction framework** with the nearest centroid taxonomy to qualify each prediction not only with a probability but also with a probability error bar
- Only real-time quantities are used
- 7 signals
- Two kinds of features per signal
  - Sampling rate: 1 kS/s
  - Time windows 32 ms long

#	Signal name	Units
1	Plasma current	A
2	Locked mode amplitude	T
3	Plasma internal inductance	
4	Plasma density	m^{-3}
5	Stored diamagnetic energy time derivative	W
6	Radiated power	W
7	Total input power	W

$$\mathbf{x} = (x_1, x_2, \ldots, x_{14}) \in \mathbb{R}^{14}$$

$$x_{2k-1} = mean(s_i(t)), \quad x_{2k} = std\left(|fft(s_i(t))|_+^*\right)$$

$$k = 1, 2, \ldots, 7$$

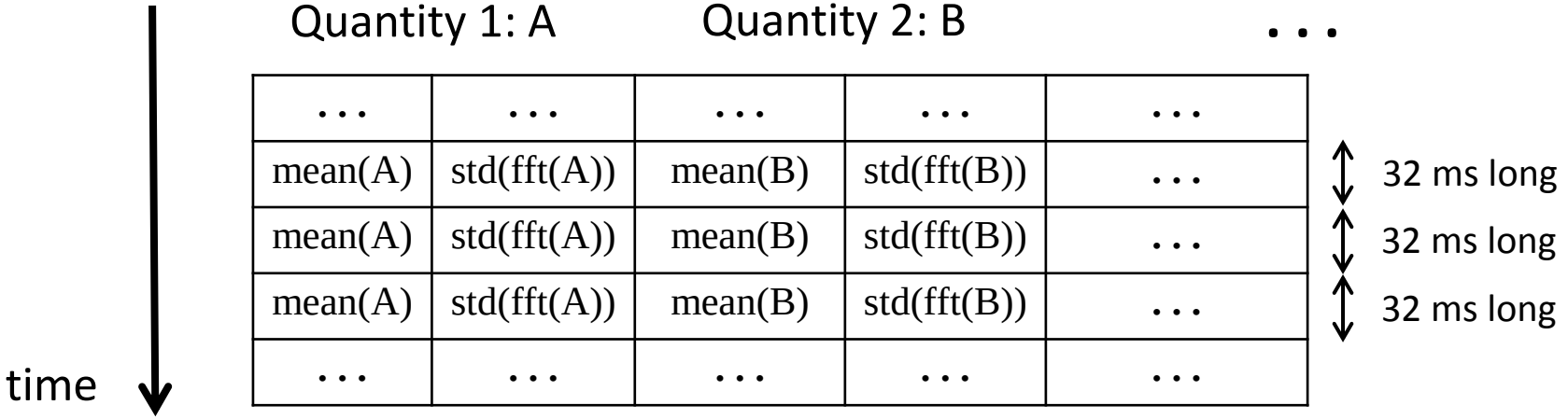
- 1237 discharges (2011-2012)
  - 1036 non-disruptive/201 disruptions
- All possible combinations with a number of features between 2 and 7 have been tested
  - 9893 different predictors have been analyzed

J. Vega, A. Murari, S. Dormido-Canto et al. "Adaptive high learning rate probabilistic disruption predictors from scratch for the next generation of tokamaks". Nuclear Fusion. 54 (2014) 123001 (17pp)

Feature vectors

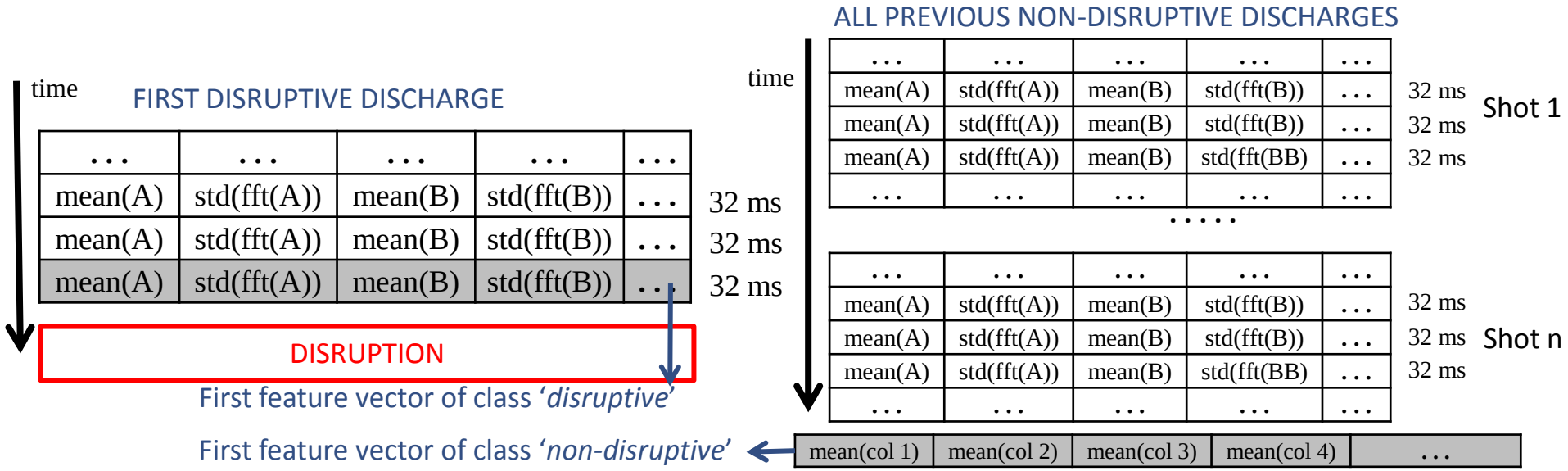
- Plasma current
- Mode lock amplitude
- Total input power
- Plasma internal inductance
- Plasma density
- Stored diamagnetic energy time derivative
- Radiated power

- Features: time windows 32 ms long
  - Time domain: mean value
  - Frequency domain: standard deviation of the power spectrum (removing DC component)
- Objective: predictors with reduced datasets of features
  - Simplicity condition: 1 single classifier with a reduced number of features



Choice of disruptive and non-disruptive training examples

- Simplicity condition: very condensed information in the feature space compatible with the best possible generalization capability
- First classifier requires 1 disruptive example and 1 non-disruptive example



- New classifiers are generated after every missed alarm
  - All feature vectors of the previous predictor
  - Last feature vector previous to the missed disruption
  - Feature vector with the mean values of all features corresponding to non-disruptive discharges from the last training
- A Venn predictor with the *nearest centroid* taxonomy has been chosen
  - To condense the information in the feature space just in two points
  - To minimise the impact of computations

JET results

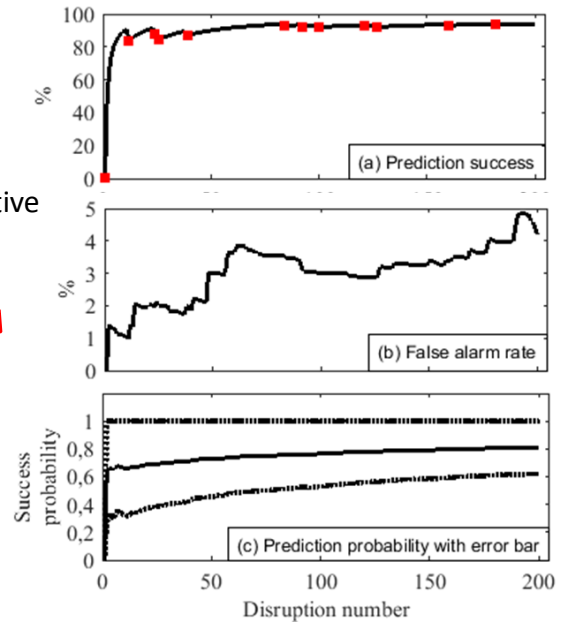
#	Signal name	Units
1	Plasma current (Ip)	A
2	Locked mode amplitude (LM)	T
3	Plasma internal inductance (Li)	
4	Plasma density (Ne)	m^{-3}
5	Stored diamagnetic energy time derivative (dW/dt)	W
6	Radiated power (Rp)	W
7	Total input power (Tp)	W

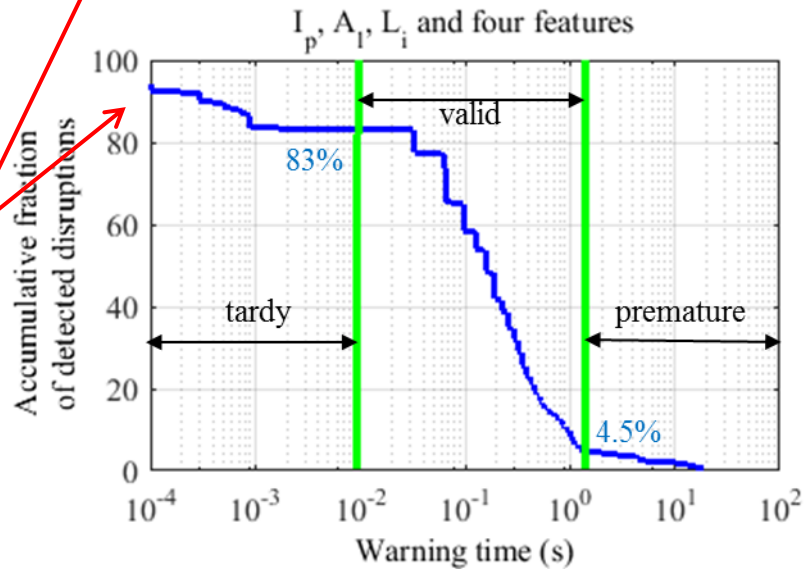
Odd numbers: time domain features $x_{2k-1} = mean\left(s_i(t)\right)$

Even numbers: frequency domain features $x_{2k} = std\left(\left|fft\left(s_i(t)\right)\right|_+^*\right)$

Ip		LM		Li		Ne		dW/dt		Rp		Tp		SR (%)	FA (%)	AVP
1	2	3	4	5	6	7	8	9	10	11	12	13	14			
		x	x											94.00	4.70	0.813±0.187
	x		x											92.50	5.09	0.831±0.169
	x	x	x											94.00	4.31	0.809±0.191
		x	x							x				94.00	4.70	0.813±0.187
	x	**x**	**x**	**x**										**94.00**	**4.21**	**0.811±0.189**
	x	x	x							x				94.00	4.21	0.810±0.190
	x	x	x	x						x				94.00	4.21	0.811±0.189
	x	x	x			x	x							94.00	4.21	0.803±0.197
	x	x	x			x	x			x				94.00	4.21	0.803±0.197
	x	x	x	x						x	x			94.00	4.31	0.810±0.190
	x	x	x	x		x	x			x				94.00	4.31	0.802±0.198
	x	x	x			x	x			x	x			94.00	4.31	0.802±0.198

1036 non-disruptive
201 disruptive





Real-time implementation of the Venn predictor

- So far, it has not been implemented in the MARTe framework

- The existing implementation has been carried out with a fast controller with DAQ FPGA-based data acquisition devices corresponding to ITER catalogue
  - In particular, a reconfigurable Input/Output platform has been used

TO BE PRESENTED

J. Vega, M. Ruiz, E. Barrera et al. “Real-time implementation with FPGA-based DAQ system of a probabilistic disruption predictor from scratch”. 11th IAEA Technical Meeting on Control, Data Acquisition, and Remote Participation for Fusion Research. May 8 – 12, 2017. Greifswald, Germany

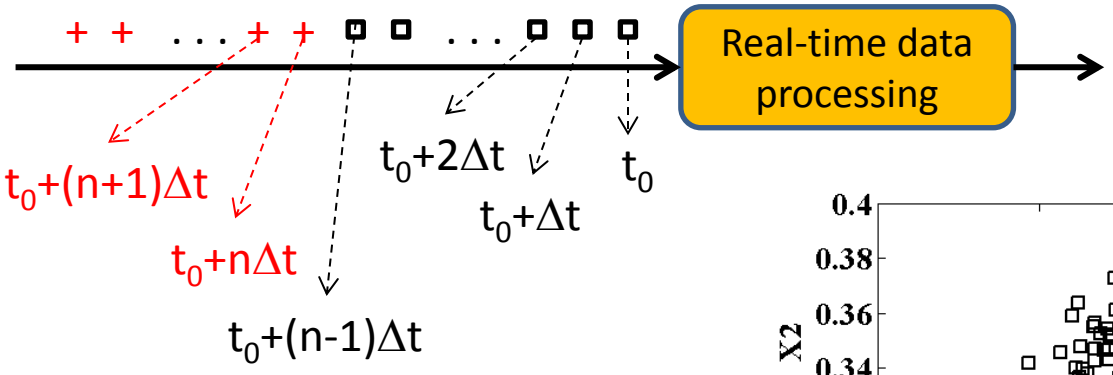
Disruption predictors based on anomaly detection

Disruption prediction based on anomaly detection

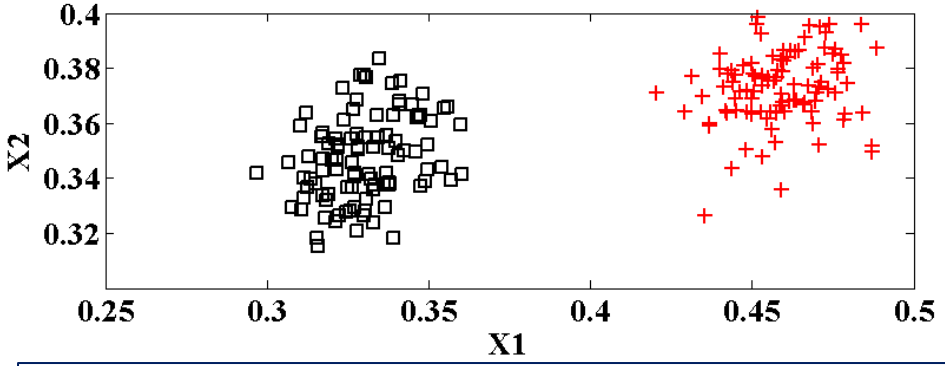
- A more advanced option for disruption prediction would be the use of intelligent predictors that start their learning process with each new discharge and *without the need of previous information from past discharges*
- These predictors can be based on the automatic recognition of changes (anomaly detections) in data streams through the identification of outliers in the data flow

Algorithm for anomaly recognition

The aim **is not** to make hypothesis testing to determine the specific distributions but recognizing asap when the sample distribution is different



The detection process has to recognize that the samples start to be '*far*' from the initial cluster and, therefore, an alarm has to be triggered

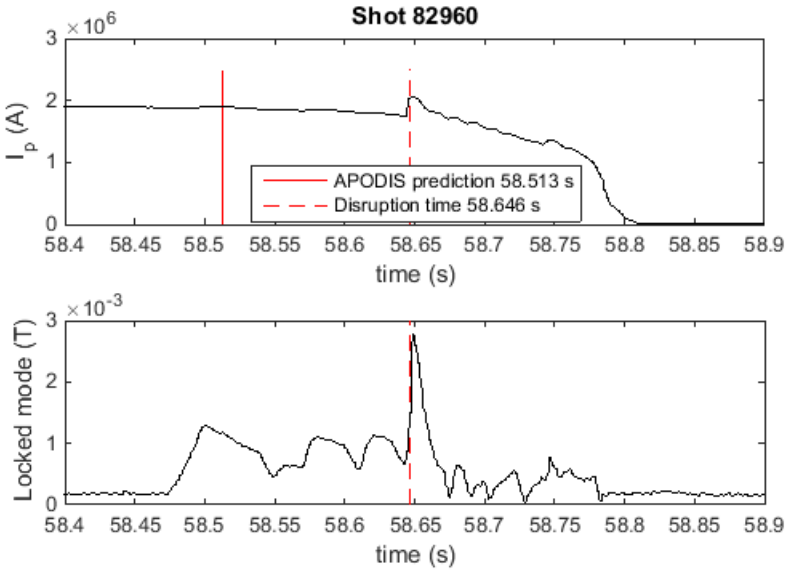


Each particular application can define a specific mapping

JET GOBIERNO DE ESPAÑA MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD Ciemat Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas

Simple Predictor based on Anomaly Detection (SPAD): on-line setting

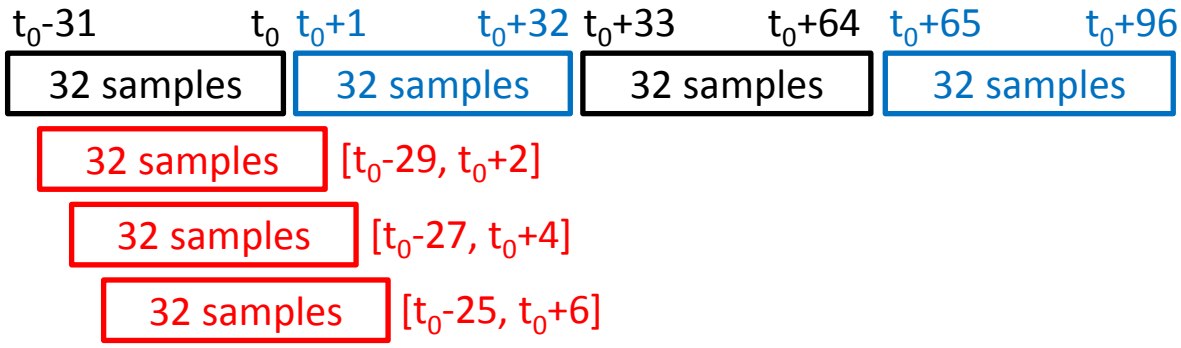
- Main advantage: no data from past discharges is needed
  - A new predictor is started with each new discharge
- Requirements
  - The delay between a true change and its detection should be minimal
  - The number of missed detections should be minimal
  - The number of false detections should be minimal
  - Data streams should be handled efficiently
  - The sequential data are read only once
- How sure are we about the fact that the anomaly detected corresponds to a disruption?
  - Feature selection to represent the plasma state is essential
  - The simplest SPAD predictor can be developed with a locked mode signal
    - Not basing the prediction only on an amplitude threshold



The JET LM predictor based on amplitude threshold misses the alarm

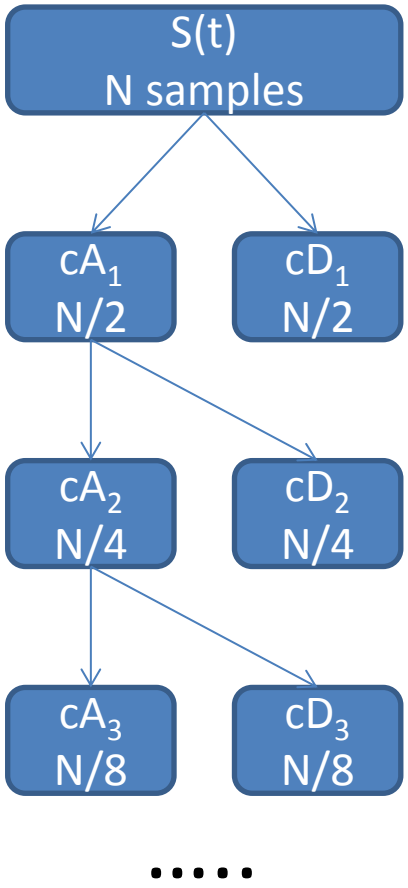
SPAD implementation

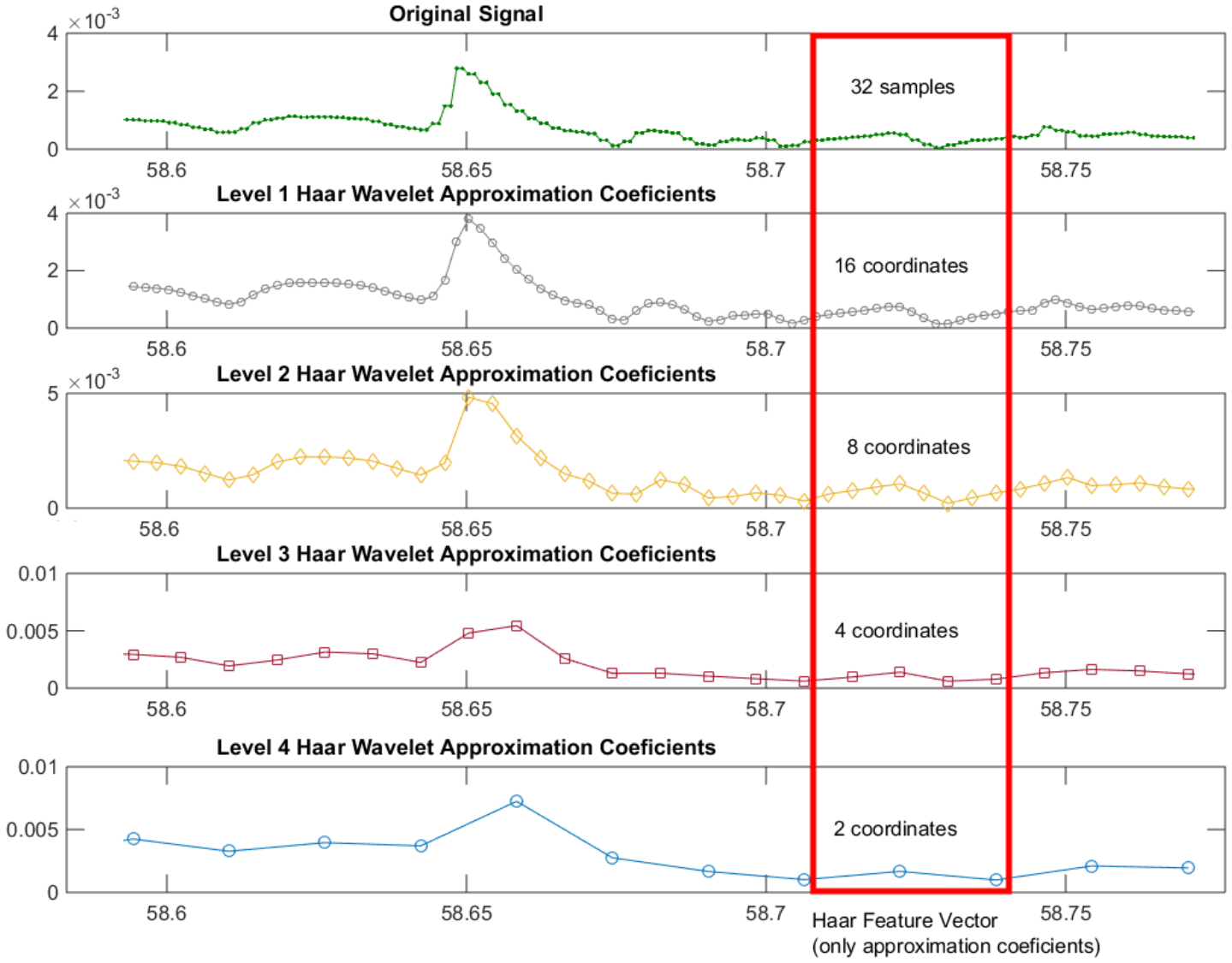
- Outlier recognition criterion
  - Data have to be processed in time windows to avoid ignoring the frequency domain
- Temporal resolution
  - Time windows: 32 ms
  - Sampling rates: 1 kS/s
  - A sliding window mechanism can be used to achieve a resolution of ms without increasing sampling rates



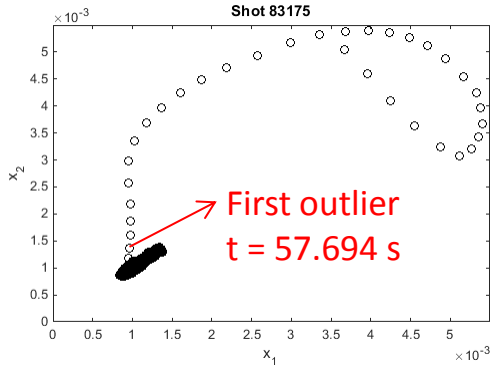
- In each time window, the data are compressed into two components by means of the Haar wavelet transform (approximation coefficients)

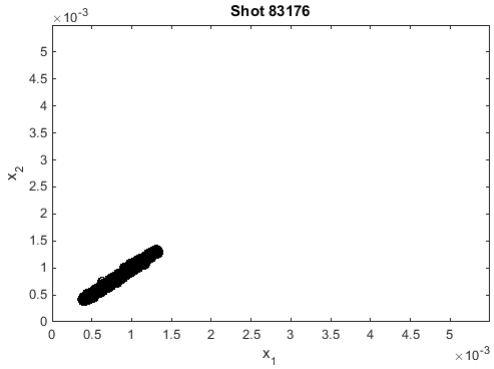
Wavelets: Haar family





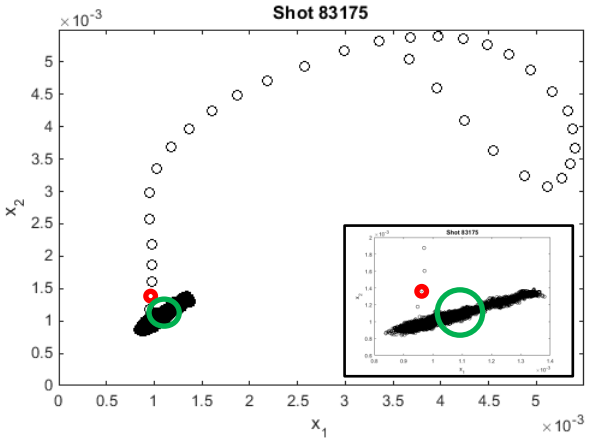
SPAD implementation





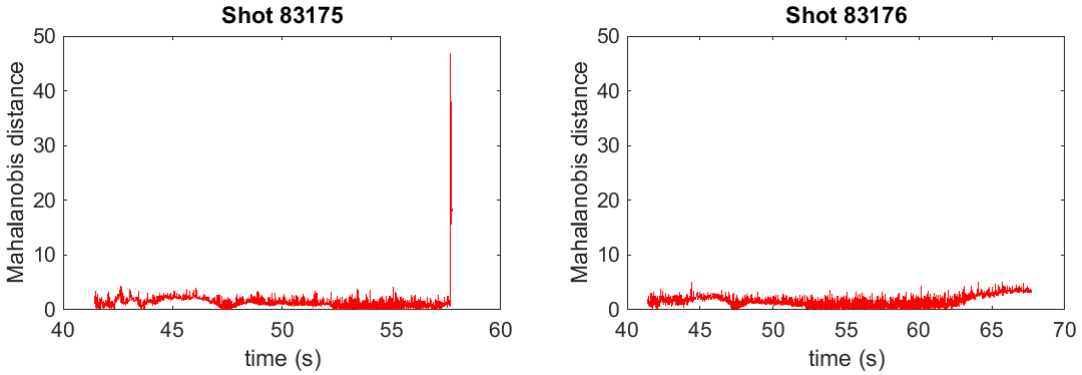
Scatterplots in the bi-dimensional space. Points are represented every 2 ms

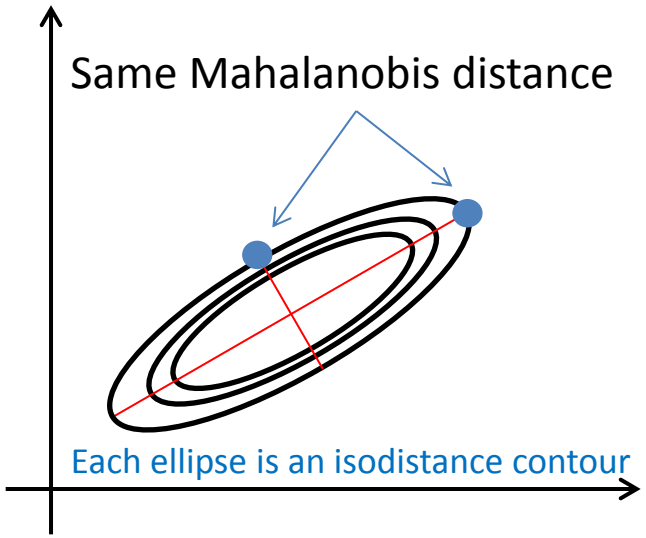
- In the non-disruptive phases of the discharges, the points in this bi-dimensional space show a compact cluster structure
- The alarm has to be triggered the first time that a point is ‘far enough’ from the cluster center
  - The Euclidean distance does not seem to work
  - The distances defined by the Euclidean metric take no account of any patterns of covariance that exist in the data
- A simple inspection of the cluster data shows a positive covariance in the data



SPAD implementation

- The metric proposed by Mahalanobis does adjust for covariance
- The computation of the Mahalanobis distance is carried out every 2 ms through the wavelet transform of the latest 32 samples of the locked mode signal



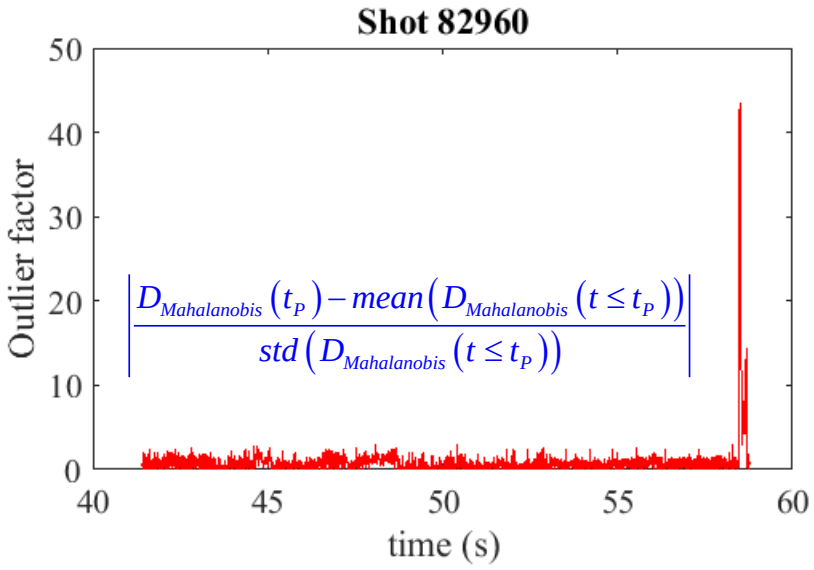


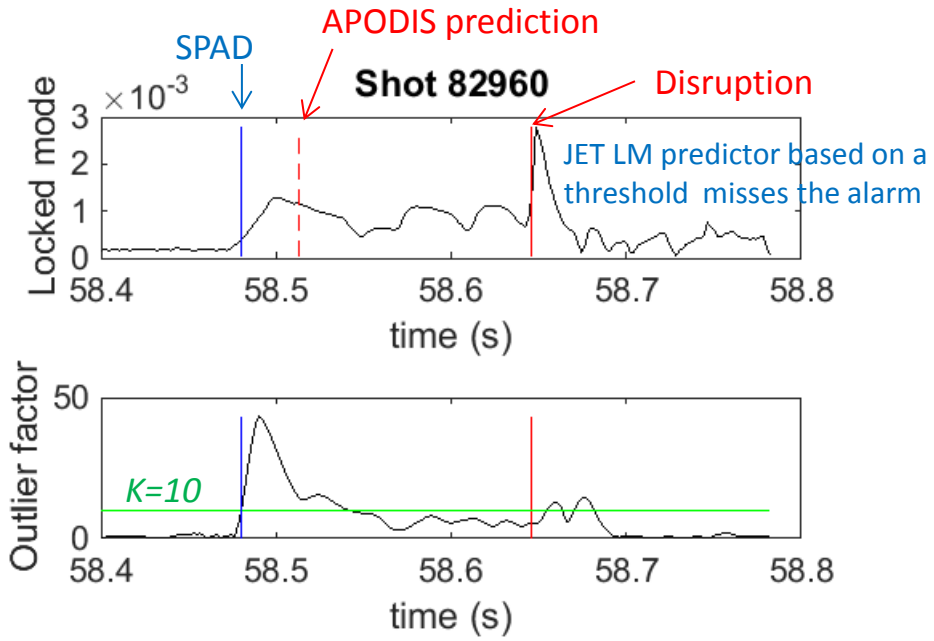
- Where is the distance limit to recognise $X(t_P)$ as outlier?
  - Outlier criterion

$$\left| \frac{D_{Mahalanobis}(t_P) - mean\left(D_{Mahalanobis}(t \le t_P)\right)}{std\left(D_{Mahalanobis}(t \le t_P)\right)} \right| \ge K$$

SPAD implementation

- In this first version $K = 10$
- In future versions, K could be determined 'on-the-fly' during each running discharge





J. Vega, R. Moreno, A. Pereira et al. "Advanced disruption predictor based on the locked mode signal: application to JET". 1st EPS Conference on Plasma Diagnostics. April 14-17, 2015. Frascati, Italy

SPAD results in JET

The information content in the 32 samples of the time windows can be compressed with the wavelet transform in 2, 4, 8 or 16 points

Data compression	False alarms (%)	Missed alarms (%)	Tardy detections (%)	Valid alarms (%)	Premature alarms (%)
2	7.13	13.43	3.53	81.45	1.59
4	7.31	11.48	3.36	83.22	1.94
8	7.42	11.84	3.00	83.39	1.77
16	+18%	12.37	3.71	81.80	2.12

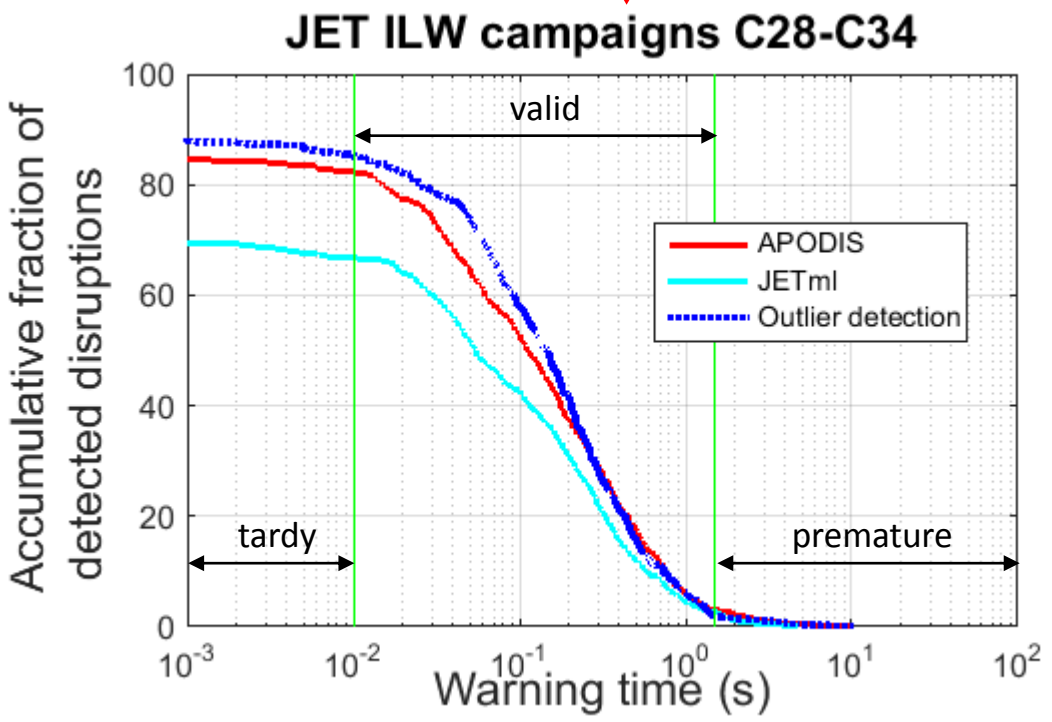
Missed alarms: no alarm or alarm triggered after the disruption

Tardy detection: warning time < 10 ms

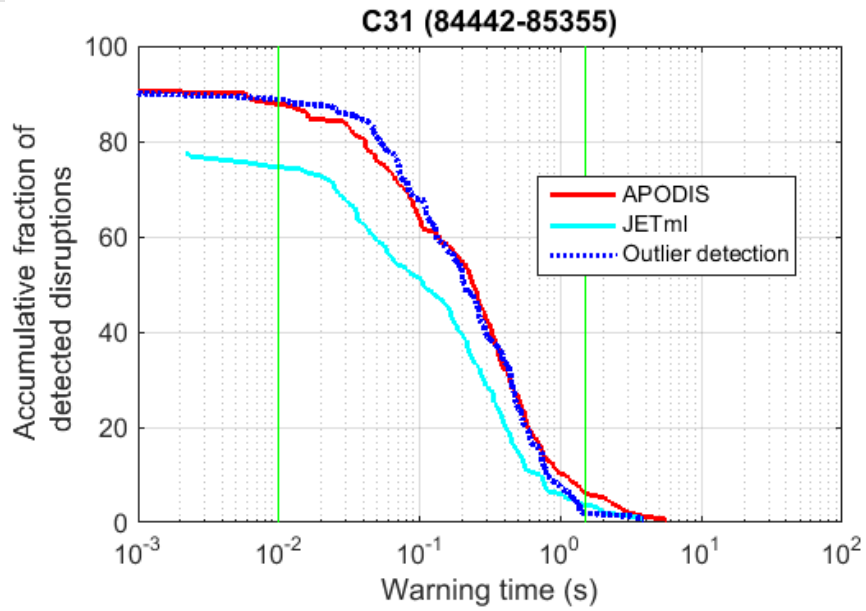
Premature alarms: warning time > 1.5 s

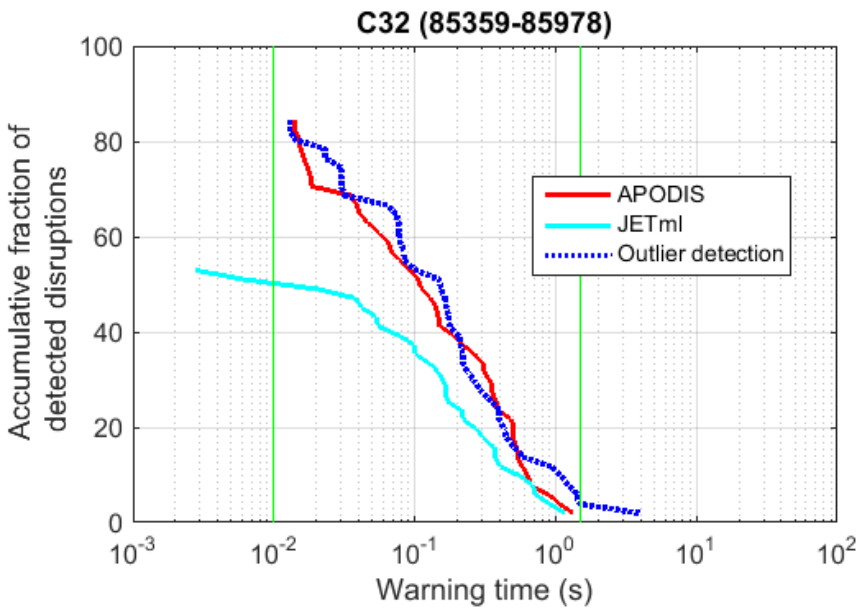
Valid alarms: 0.01 s ≤ warning time ≤ 1.5 s

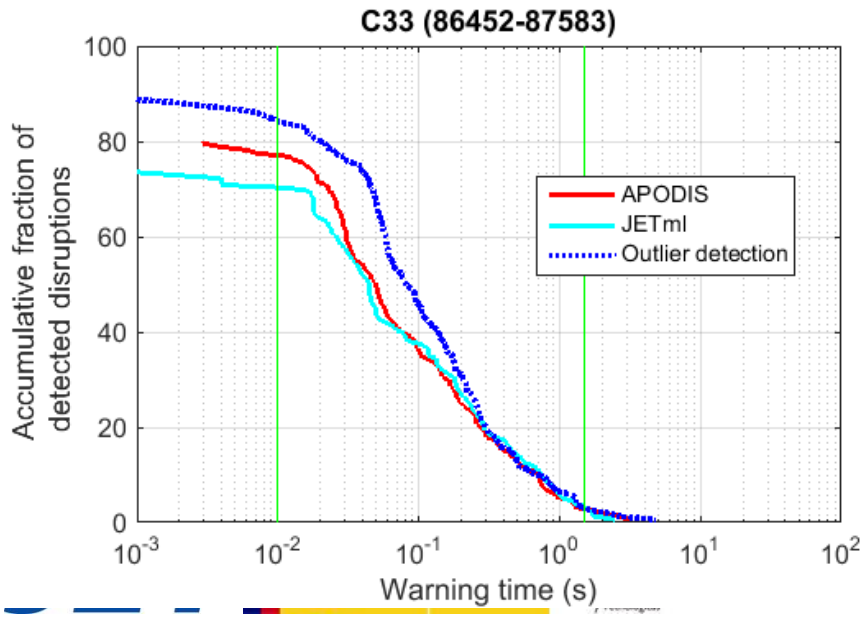
- All safe discharges and all unintentional disruptions during the ILW experimental campaigns 2011-2014 have been considered
  - 1738 non-disruptive discharges
  - 566 unintentional disruptive discharges

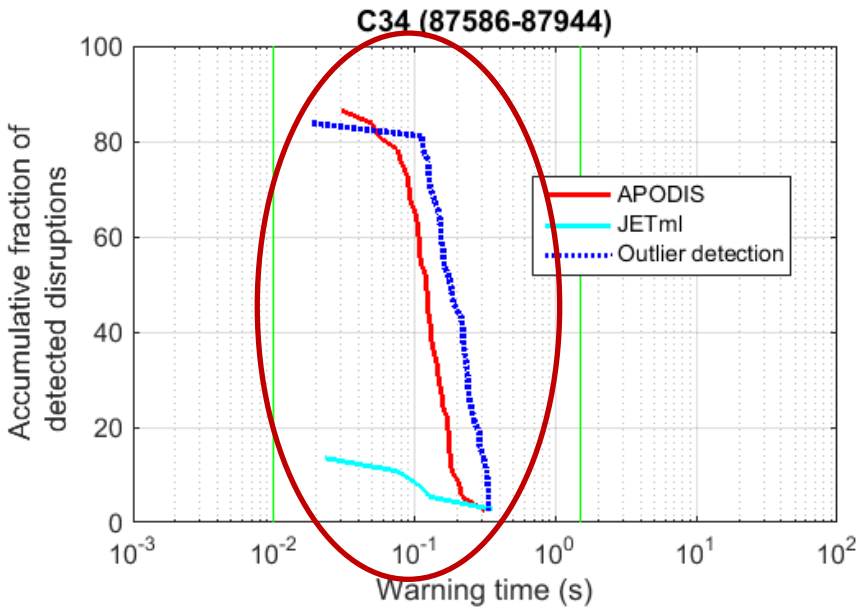


SPAD results by campaigns: July 2013 – September 2014

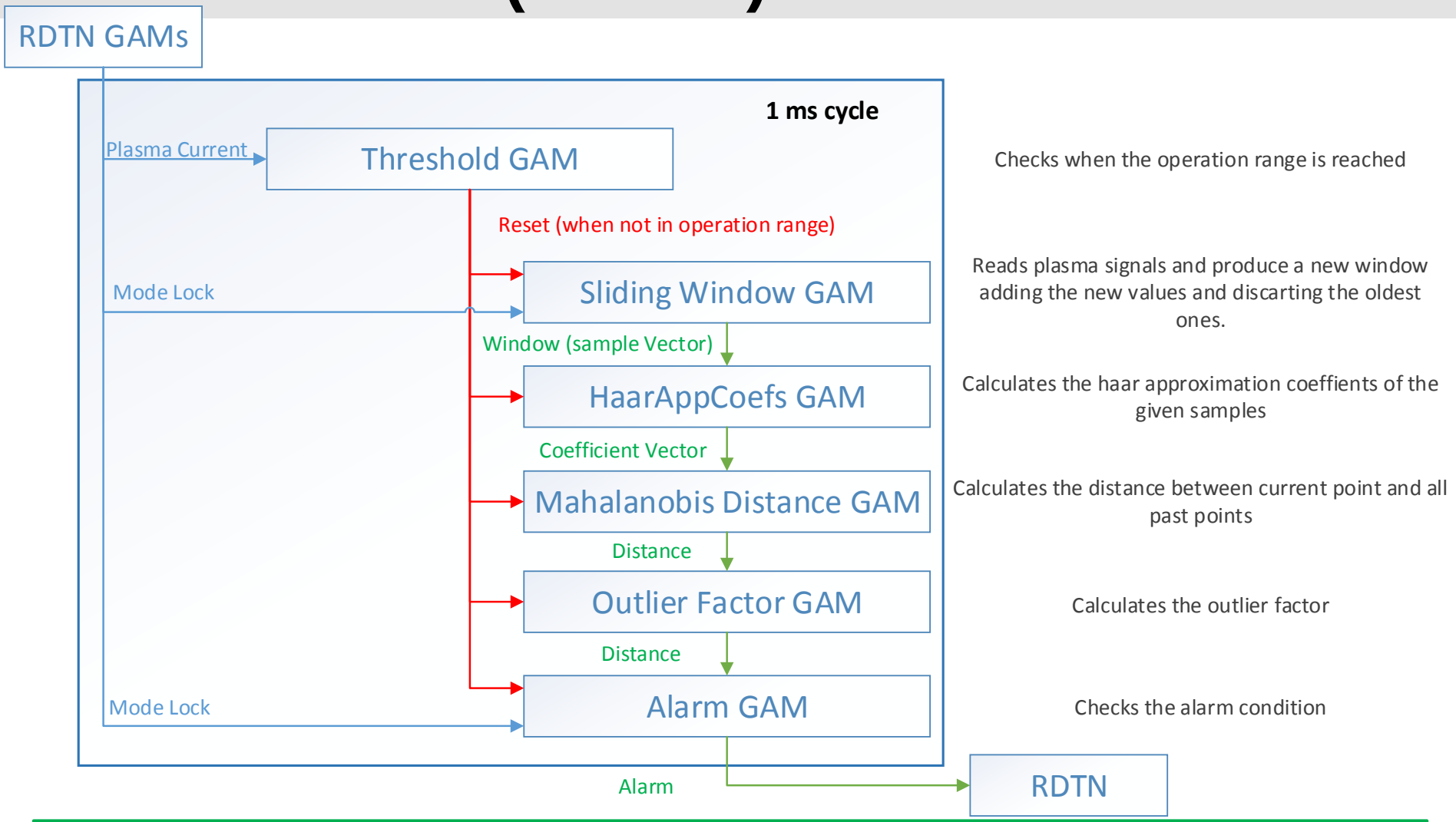








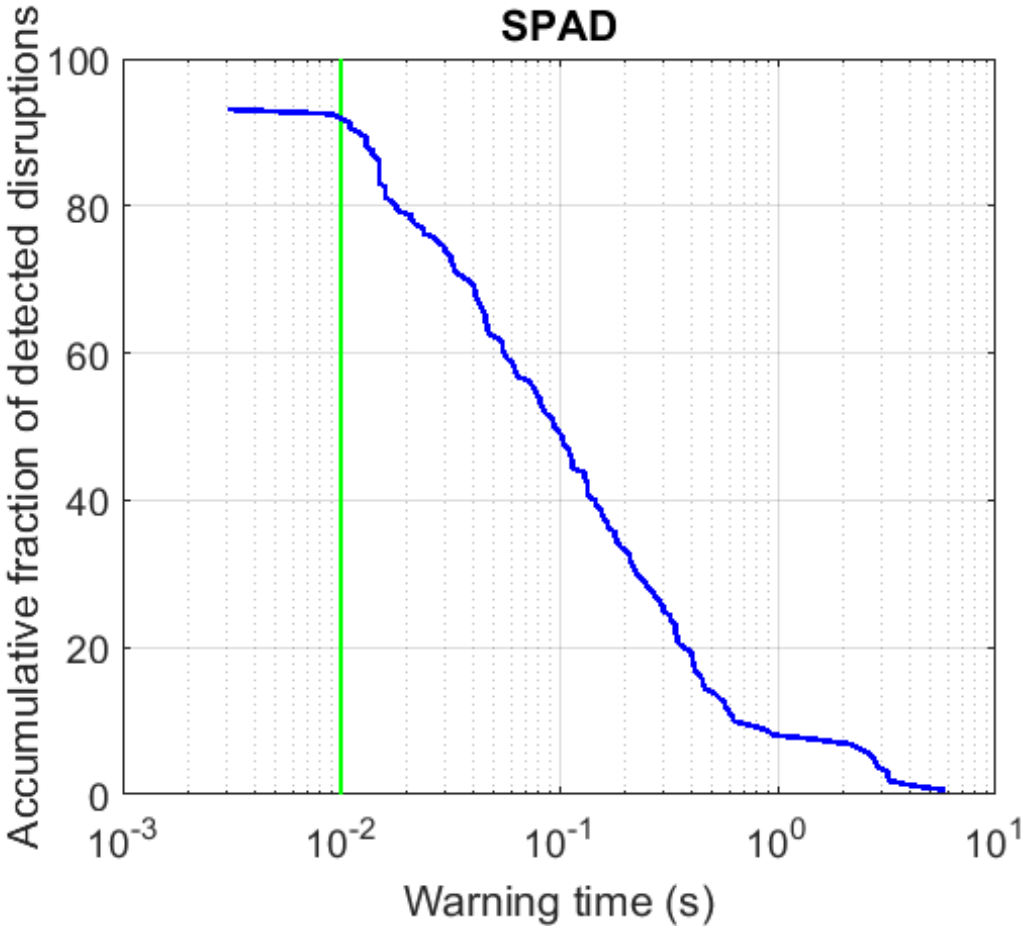
Implementation in the JET real-time data network (MARTe)



S. Esquembri, J. Vega, A. Murari et al. "Real-time implementation in JET of the SPAD disruption predictor using MARTe". Conference Record of the 20th IEEE Real-Time Conference. Jun 5th-10th, 2016. Padova, Italy.

Results of the RT implementation

- SPAD is in operation in the RT network from discharge 91807
  - Shots 91807-92504 (2016)
    - 160 disruptive
    - 387 non-disruptive
  - Success rate: 92%
  - Tardy detections: 1%
  - False alarms: 21%



Average warning time: 401±862 ms

Summary

- The prediction of disruptions is strongly related to the recognition of precursors in plasma signals
  - Disruption prediction is crucial to put into operation mitigation actions
- The combination of the time and the frequency domains greatly improves the recognition of precursors
  - Data-driven predictors do not allow recognising the physics reasons but ensure the triggering of mitigation actions for most disruptions and with enough warning time
- The combination of the time and the frequency domains avoids the 'ageing' effect associated to features in the time domain
- This presentation has reviewed the evolution of statistical learning methods during the last years for the real-time prediction of disruptions in JET
  - '*Classical*' approaches: the more training samples the better
  - *Prediction from scratch*: adaptive learning predictors with very limited number of training examples
  - Predictors based on *anomaly detection*: no data from past discharges are needed
- Adaptive predictors will be always necessary either in the '*prediction from scratch*' approach or in an '*anomaly detection*' approach
  - New knowledge has to be incorporated as it is necessary