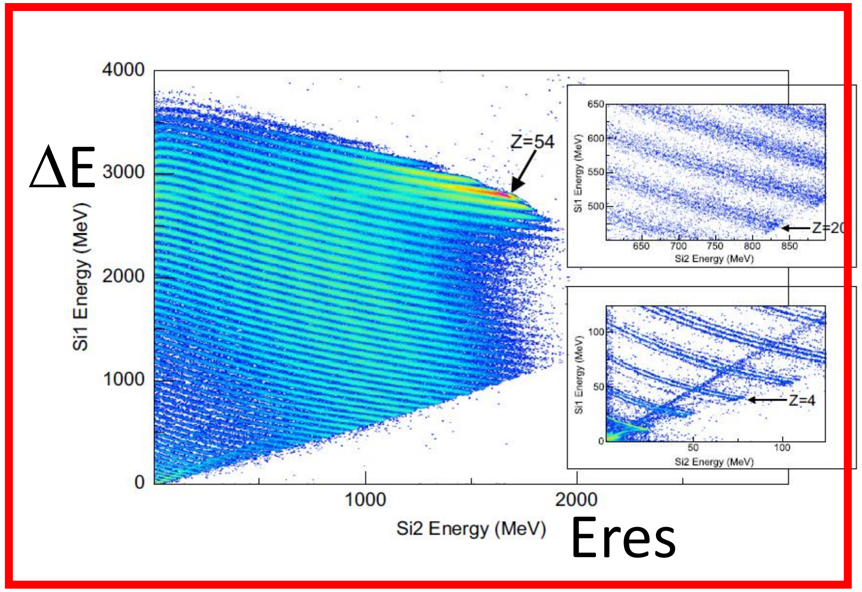
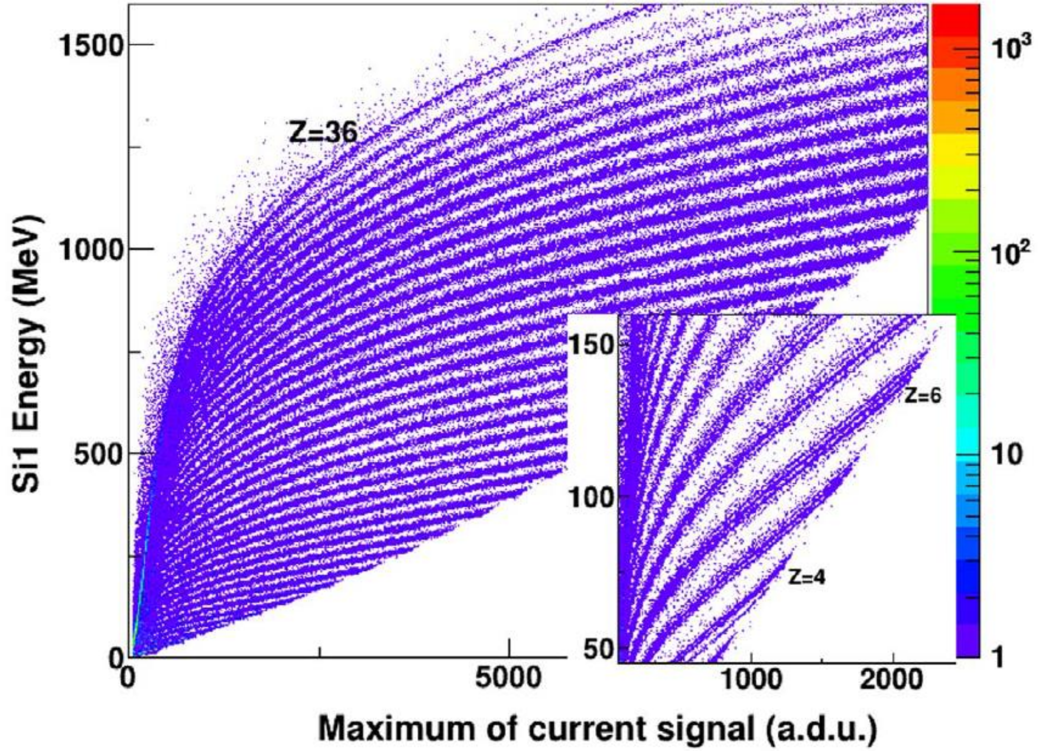
PSA identification by means of Machine Learning: a first attempt

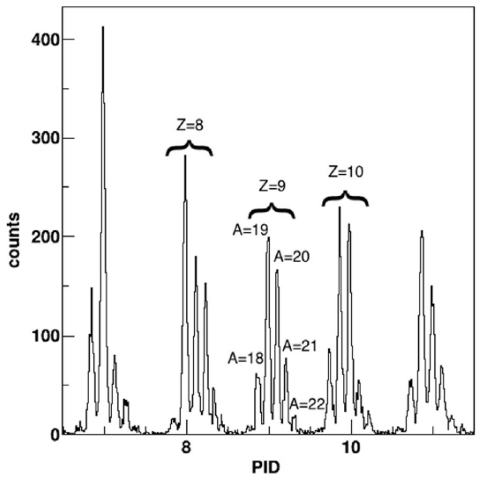
Silvia Piantelli – INFN Sezione di Firenze

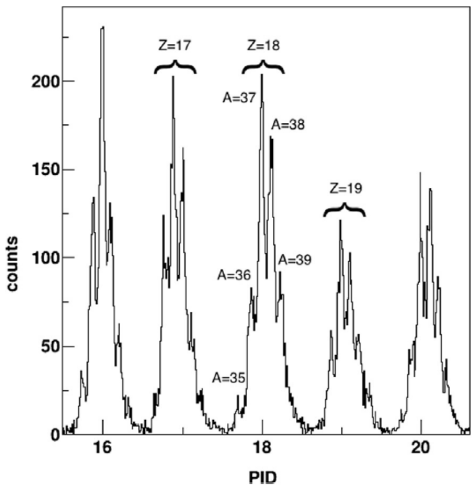
The problem: Z and A identification for particles stopped in Si1

- At present there is not a technique to calculate the lines as it can be done for Si-Si (and Si-CsI) correlation (i.e. Bethe Block formula)
- The current method: clicking by hand
- Is it possible to automatize the procedure?
- Can machine learning help? Maybe…







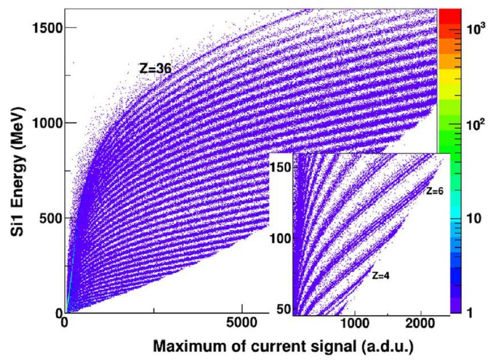


First attempt: identification from I_{max}-E matrix

- Used package: python3 with numpy, pandas, keras and tensorflow
- For one detector, for which the PID had been already done, a csv file with E,I,Z for each point on the 2D matrix was produced (including not identified ions)

	Imax	E	z
0	7065111.0	75.423126	8.0
1	2581548.0	36.507473	-1.0
2	38831628.0	123.455101	7.0
3	77587944.0	182.867996	8.0
4	2848768.0	28.826826	5.0

- The dataset was divided in training part (80%) and test part (20%)
- I and E values were normalized to 1
- It's a classification problem, with supervisioned training
- The adopted architecture is 4 Dense layer, 20 training epochs
- The result was reasonable: the confusion matrix gives 96% accuracy
- Not so bad, but it is difficult to think to export this neural network to other detectors, once it is trained on one detector

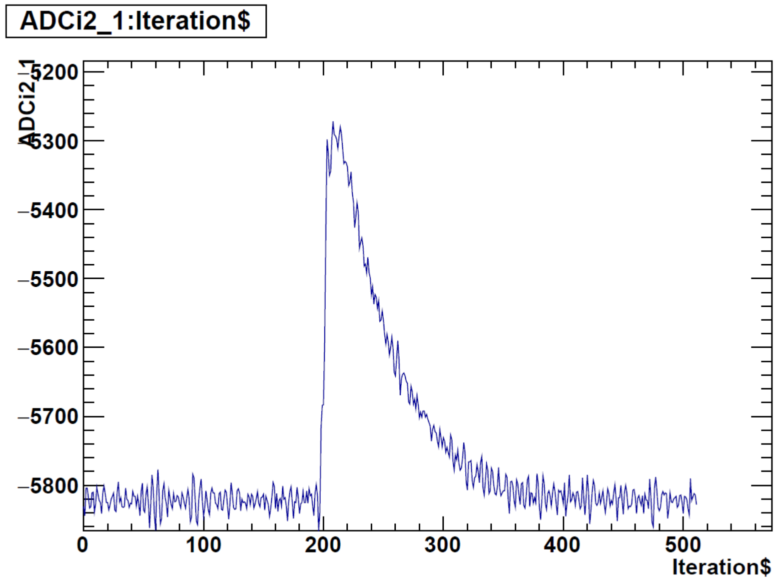


Confusion matrix

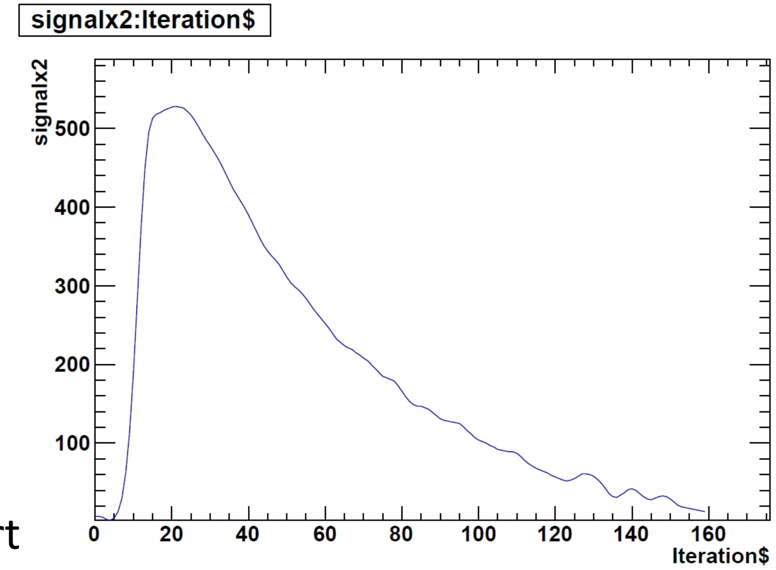
```
[[2786  103   63   32   14  151   87    7   42    0    0    0]
 [   0 5198    6    0    0    0    0    0    0    0    0    0]
 [   0    0  574    5    0    0    0    0    0    0    0    0]
 [   0    0    0  549    0    0    0    0    0    0    0    0]
 [   1    0    0    1 2183    0    0    0    0    0    0    0]
 [   0    0    0    0    0 5901   23    0    0    0    0    0]
 [   3    0    0    0    0    0 6076  129    0    0    0    0]
 [   0    0    0    0    0    0    2 6967   44    0    0    0]
 [  41    0    0    0    0    0    0    9 3453   77    0    0]
 [  16    0    0    0    0    0    0    0   14 1840   52    5]
 [   3    0    0    0    0    0    0    0    0    0  178    8]
 [   0    0    0    0    0    0    0    0    0    0  380   73]]
```

Second attempt (still in progress, very preliminary results): identification from the signal shape

- The signal shape should contain much more information than I_{max} and E => maybe it is possible to train the network on one detector and then export it to others
- Step 1: pre-processing signals



Base line removal
Smoothing
Cut of the not significant part



NB for this detector (FAZIASYM data) the Z,A value is associated to each signal shape, but at present not identified particles are not included in the dataset (but it is important to include them too)

- Normalization to 1 of the maximum amplitude

Second attempt (still in progress, very preliminary results): identification from the signal shape

- Step 2: the dataset was divided in a training part (80%) and in a test part (20 %)
- Step 3: choice of the architecture
  - It it known that for this kind of problems it is better to have some 1D convolutional layers followed by Dense layers
  - An attempt done in such a way did not produce good results (the network did not train)
  - A very expert people of INFN Firenze suggested:
    1. Start with only Dense layers and then try to add 1D Conv one by one
    2. change the cost functions (with respect to my first choice)
    3. Change the optimizer
    4. Instead of doing a classification problem, try with a regression problem (in progress), i.e. you obtain an intepolated value (like a PID) not a Z value
    5. Try to reproduce also the energy (still not done)

With the Dense layers only the network can be trained
Training on 500 epochs and only 80 signals
Accuracy 0.55
Confusion matrix (test on 20 signals)

[[0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0]
[0 1 0 0 0 0 0 0 0 0]
[0 0 0 1 0 0 0 0 0 0]
[0 0 0 0 2 0 0 0 0 0]
[0 0 0 0 0 0 2 0 0 0]
[0 0 0 0 0 1 0 0 0 0]
[1 0 0 0 0 0 0 3 2 0]
[0 0 0 0 0 0 0 1 4 1]
[0 0 0 0 0 0 0 0 0 1]]

Next steps

- Regression network
- Test with 1DConv adding the layers step by step
- Test on all the statistics
- Extension of the model to other detectors with and without training on their data
- Test on the same detector in different conditions (radiation damage, mounting at other angles, etc.)
- Futuristic developments: implementation on a FPGA (a simplified network, keeping the parameters of a complete one trained off line)