UPV now comes with many utilities

UPV

h(t) veto production

Channels are processed one by one and vetoes are produced using the trigger frequency

Online vetoes

The architecture to produce vetoes based on UPV thresholds is now in place

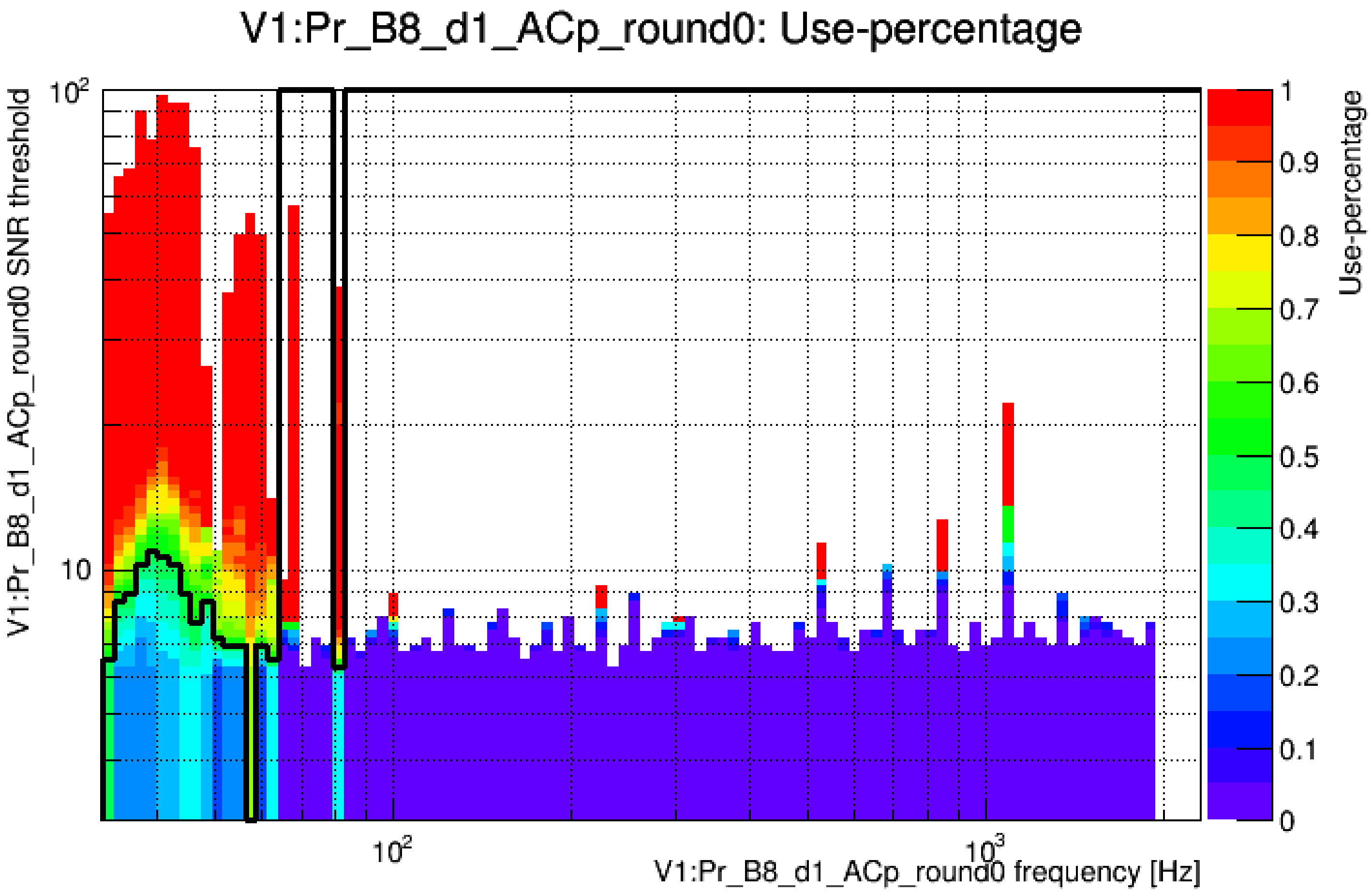
h(t) hierarchical veto

Same approach as hveto:
- redundancies are removed
- dead-time is optimal

Matrix

Every pair of channels is considered → very useful to characterize noise paths and couplings

UPV veto production



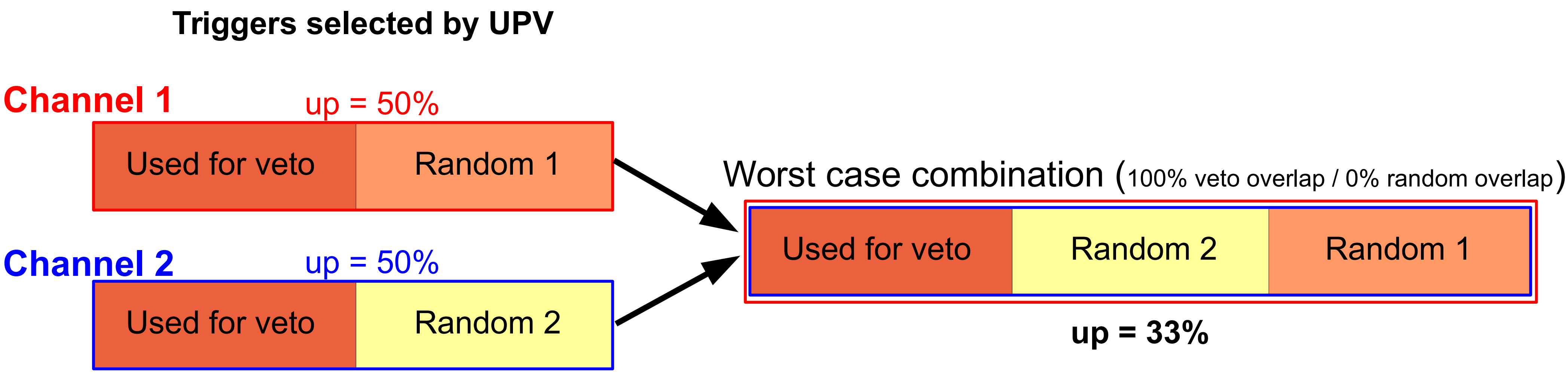
A threshold is adjusted as a function of the trigger frequency to provide optimal veto performance

UPV hierarchical approach

Why do we need a hierarchical approach?

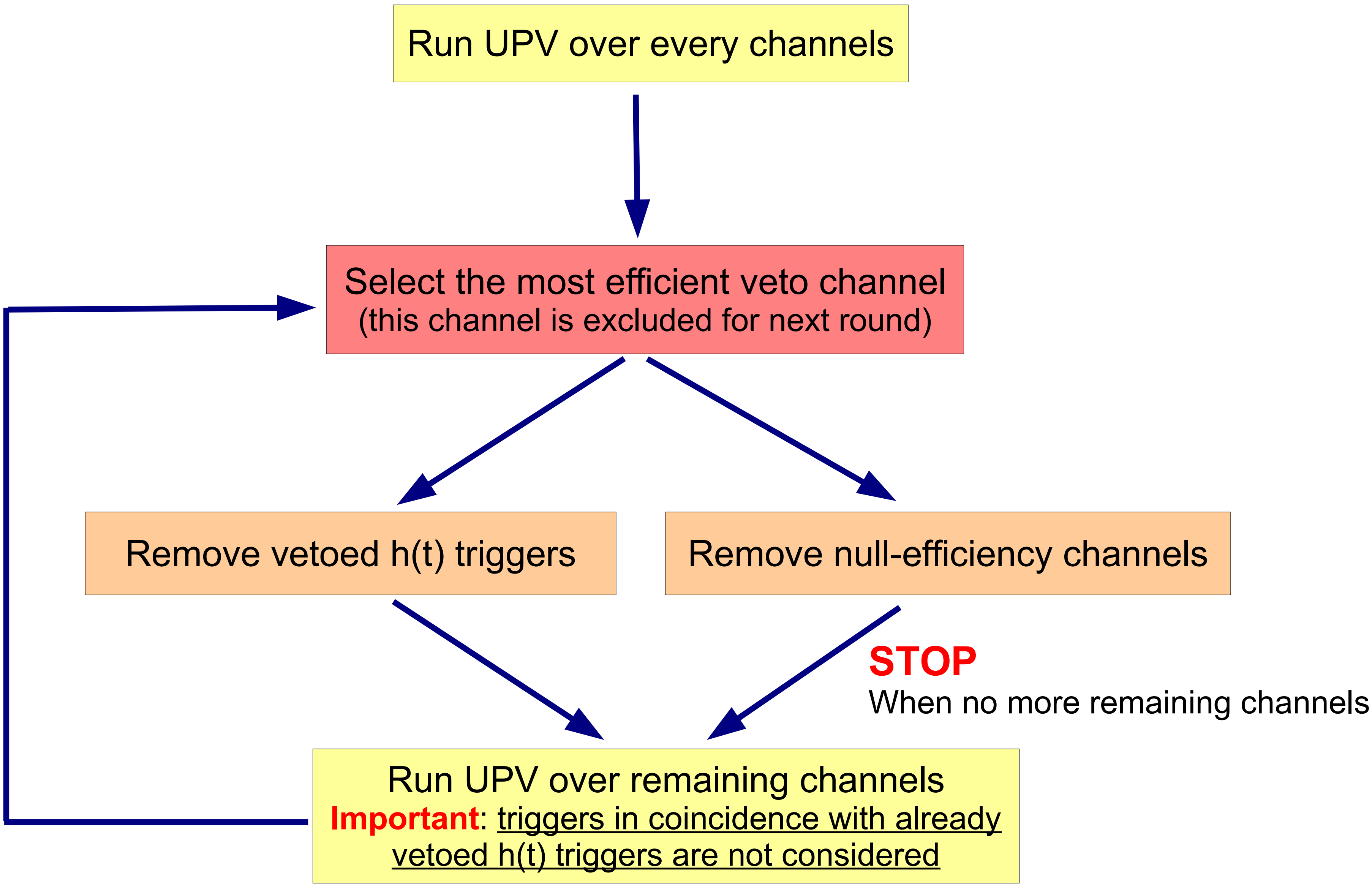
UPV is run on several hundreds of channels. Many of them provide redundant data quality information.

We cannot blindly combine hundreds of vetoes. The channel-by-channel performance is very good but the overall performance is poor.

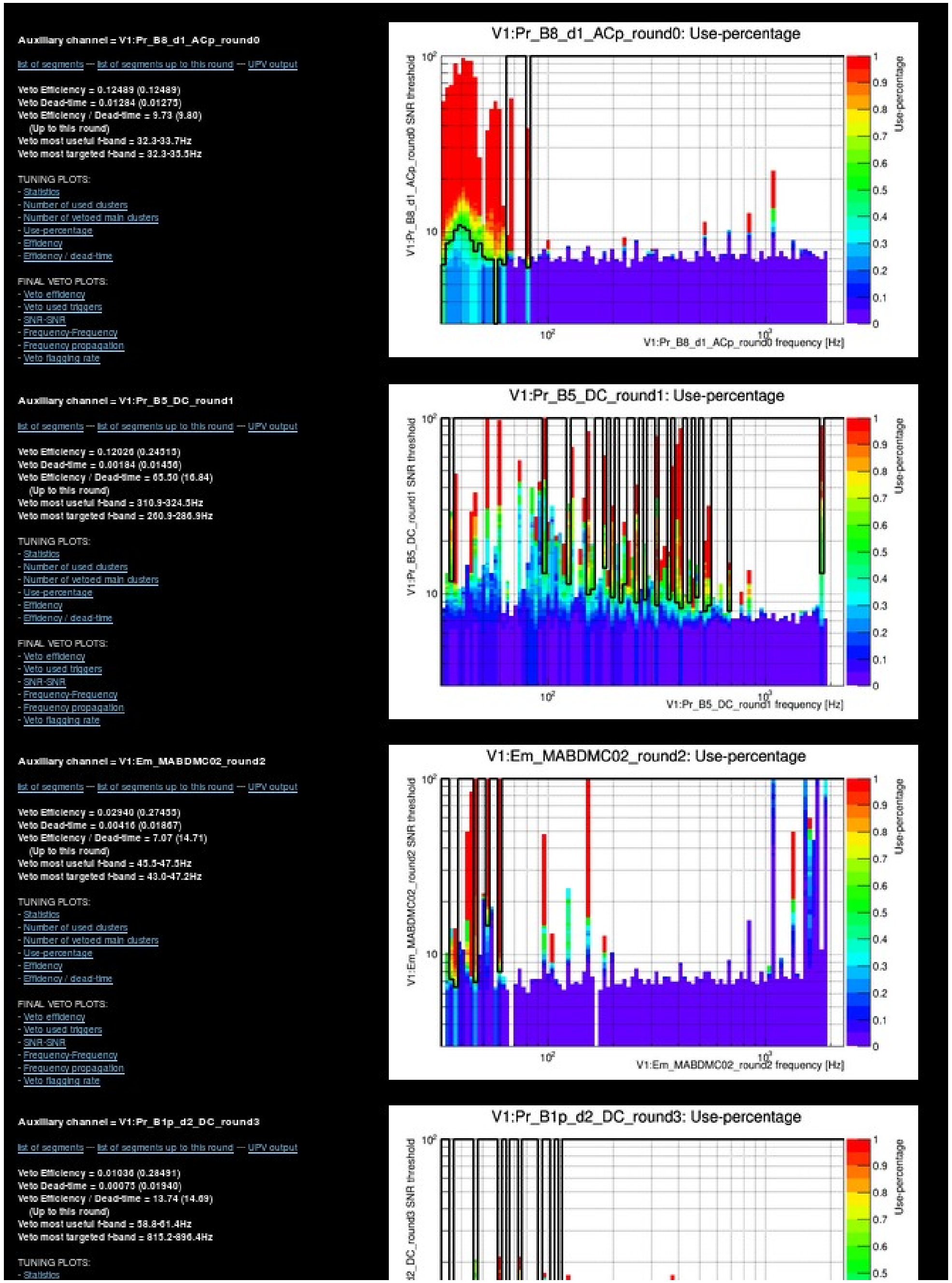


→ We need to combine a minimal number of channels

UPV hierarchical algorithm



UPV hierarchical report



UPV report round by round

The hierarchical approach of UPV has many advantages:

– A minimal set of channels are selected for vetoing

– The dead-time is minimal

– Channels are naturally ranked → helpful for noise investigations

Preliminary tests on VSR2 data show good performance:

For h(t) triggers with SNR>8:
- number of channels = 13
- dead-time = 1.2%
- efficiency = 48%

UPV online

The architecture to produce online vetoes based on UPV optimized thresholds is now in place

• Thresholds are produced by UPV and saved on disk (ROOT files). We need to determine how often this tuning should be performed

• Thresholds are loaded and veto segments are produced by Omicron when the triggers are produced (latency~20s)

• Veto segments are saved in a FrEvent structure and send to SegOnline

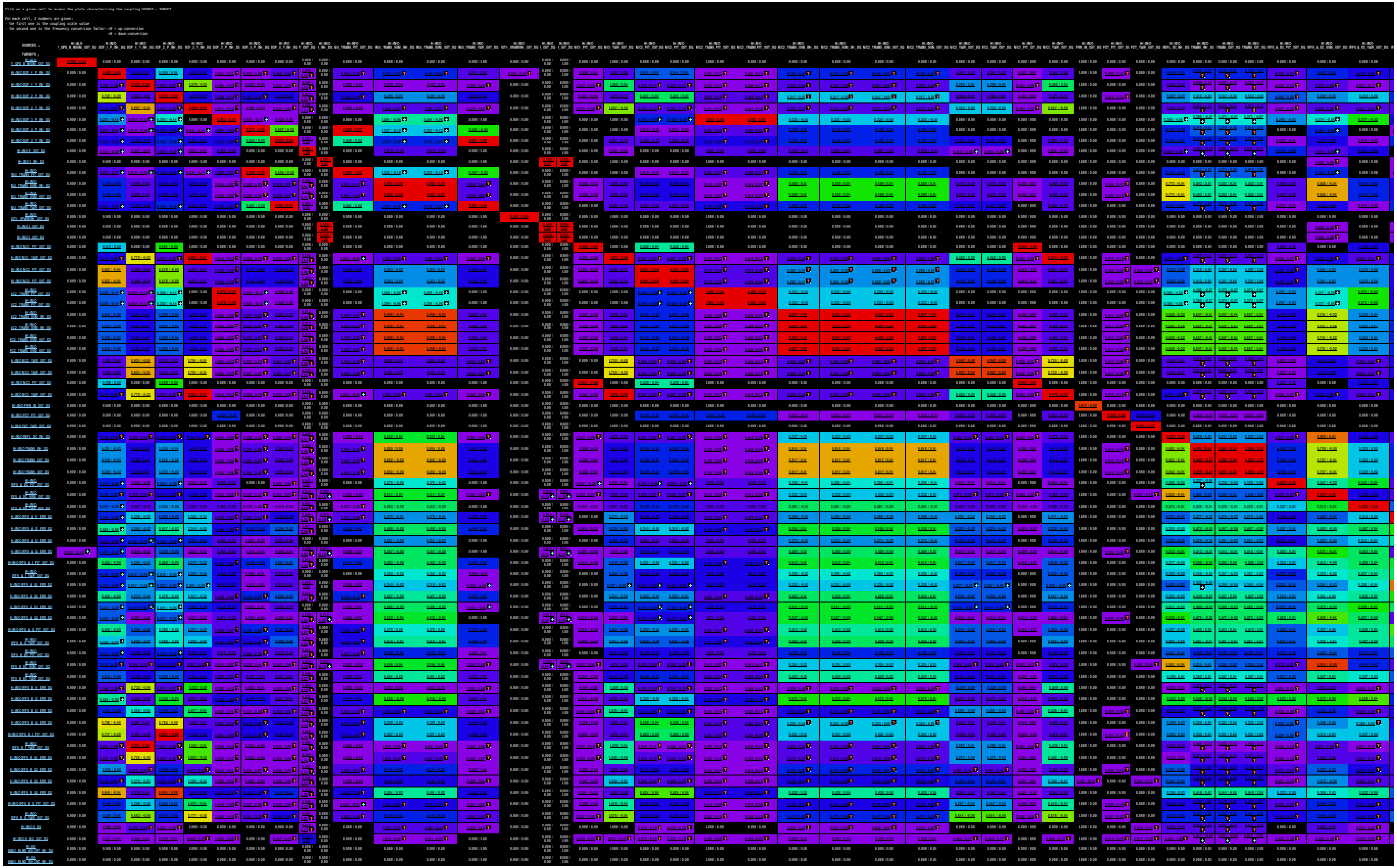
• Threshold files can be updated anytime without stopping any process

→ Some preliminary tests were successful. More tests (greater scale) are needed.

→ Some threshold history needs to be implemented

→ We need to think about about a way to save these vetoes (I'm pessimistic about DQSEGDB)

UPV matrix



UPV matrix

SOURCES → TARGETS ↓	H1:ALS-Y_QPD_B_NSUM_OUT_DQ	H1:IMC-DOF_1_P_IN1_DQ	H1:IMC-DOF_1_Y_IN1_DQ	H1:IMC-DOF_2_P_IN1_DQ	D
H1:ALS-Y_QPD_B_NSUM_OUT_DQ	0.998 / 0.00	0.000 / 0.00	0.000 / 0.00	0.000 / 0.00	
H1:IMC-DOF_1_P_IN1_DQ	0.000 / 0.00	0.992 / 0.00	0.110 / 0.01	0.300 / 0.01	
H1:IMC-DOF_1_Y_IN1_DQ	0.000 / 0.00	0.086 / 0.09 ↑	0.995 / 0.00	0.064 / 0.07 ↑	
H1:IMC-DOF_2_P_IN1_DQ	0.000 / 0.00	0.720 / -0.00	0.092 / -0.01	0.993 / 0.00	
H1:IMC-DOF_2_Y_IN1_DQ	0.000 / 0.00	0.106 / 0.07 ↑	0.807 / -0.01	0.093 / 0.06 ↑	
H1:IMC-DOF_3_P_IN1_DQ	0.000 / 0.00	0.250 / -0.11 ↓	0.044 / -0.12 ↓	0.393 / -0.11 ↓	
H1:IMC-DOF_4_P_IN1_DQ	0.000 / 0.00	0.078 / -0.07 ↓	0.019 / -0.06 ↓	0.143 / -0.08 ↓	
H1:IMC-DOF_4_Y_IN1_DQ	0.000 / 0.00	0.129 / -0.05 ↓	0.134 / -0.04	0.162 / -0.06 ↓	
H1:IMC-F_OUT_DQ	0.000 / 0.00	0.012 / -0.09 ↓	0.017 / -0.01	0.058 / -0.07 ↓	

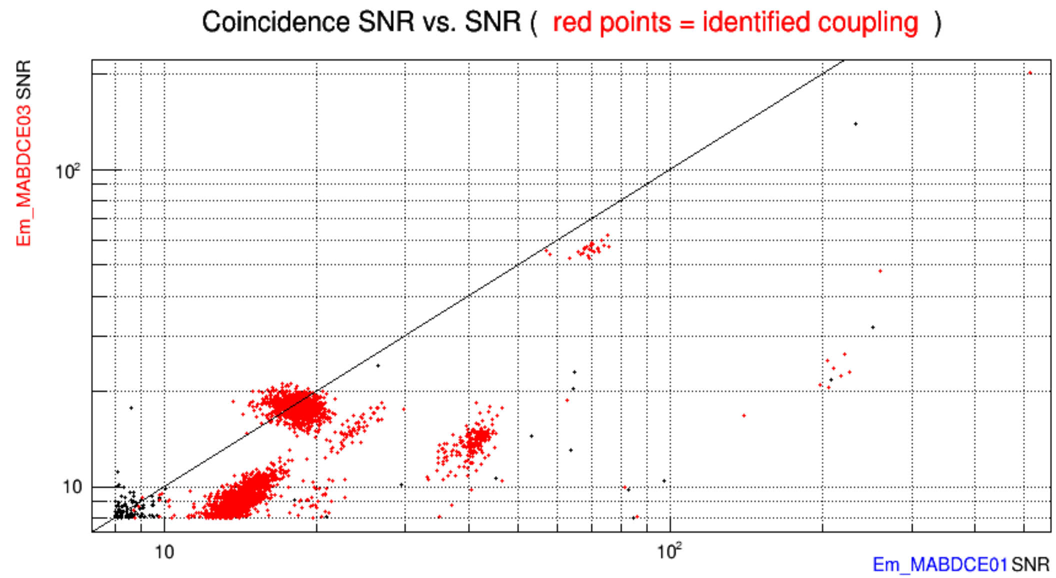
Coupling scale (~efficiency)

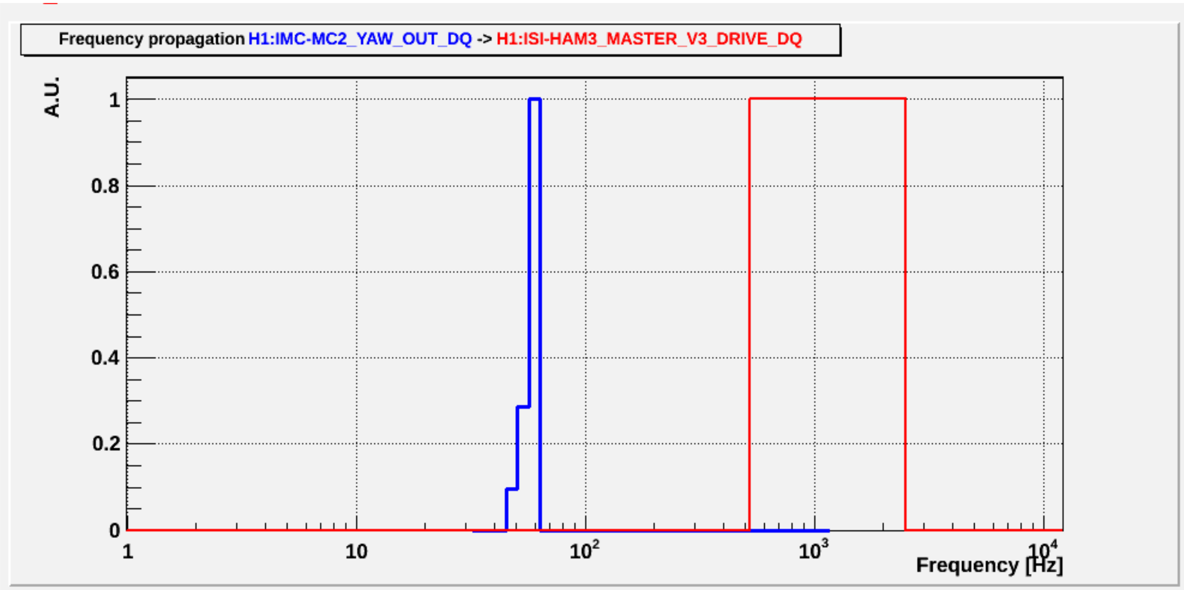
Conversion factor
>0 → up conversion
<0 → down conversion

up/down conversion indicator

UPV matrix

Diagnostic plots are provided for each identified coupling





UPV conclusions

- The UPV algorithm can be used for many purposes (veto, online, detchar...)
- UPV is being used in LIGO (N. Christensen) → more and more stable/reliable
- UPV will be a useful tool for Virgo detchar
- UPV is user-friendly: one command line → analysis + web report
- A paper is in preparation (co-written with Nelson)