



LArTPC Event Display Ruminations

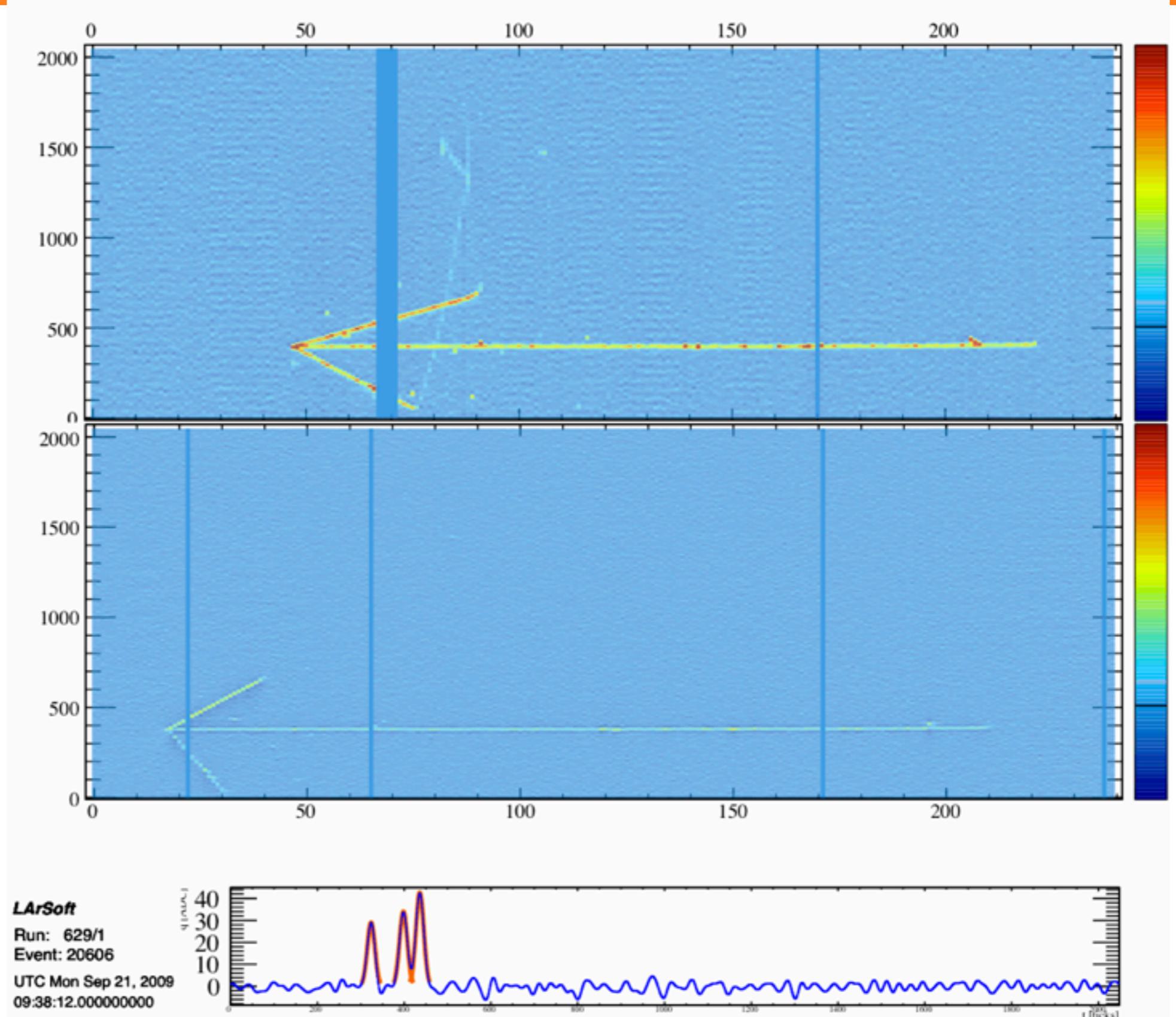
Mitch Soderberg

Aug. 17, 2011

Introduction

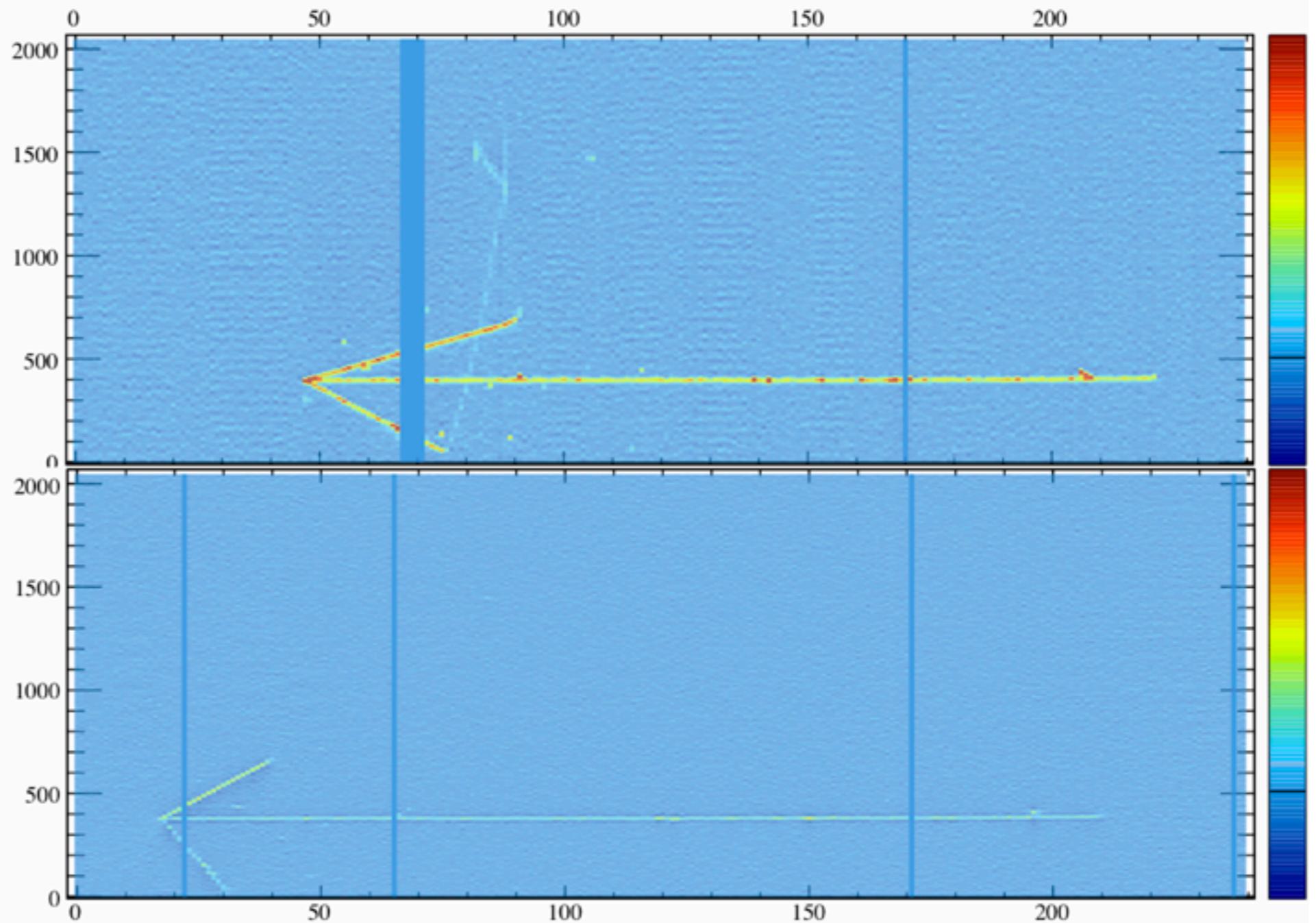
- The following is a quick update on a recent change I made to the EventDisplay, and then a few slides to instigate discussion about a “feature” inherent to the existing display.
 - ▶ The recent change: displaying of Hit objects on wire signal.
 - ▶ The discussion: Displaying very large amounts of data (i.e. - ArgoNeuT/ MicroBooNE TPC views) in ROOT.

Event Display with Hits Overlaid



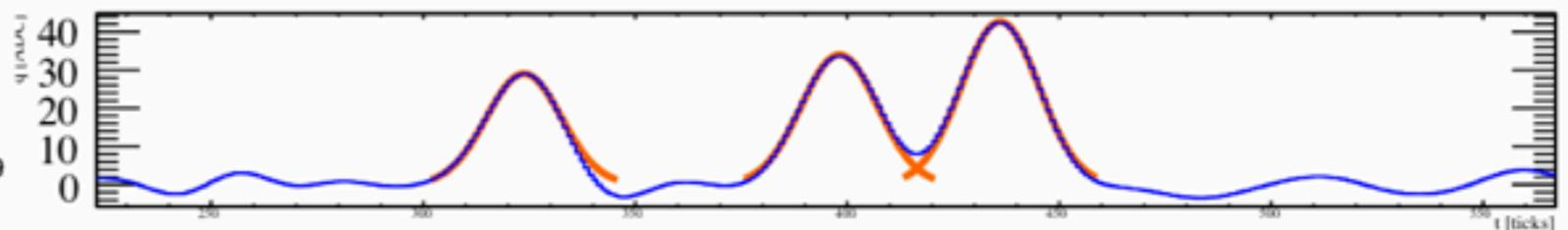
Use Hit parameters to
create Gaussian
TPolyLines.

Event Display with Hits Overlaid



Gaussians are drawn
5-sigma wide.
(Hit object only
extends from -1,+1
sigma.)

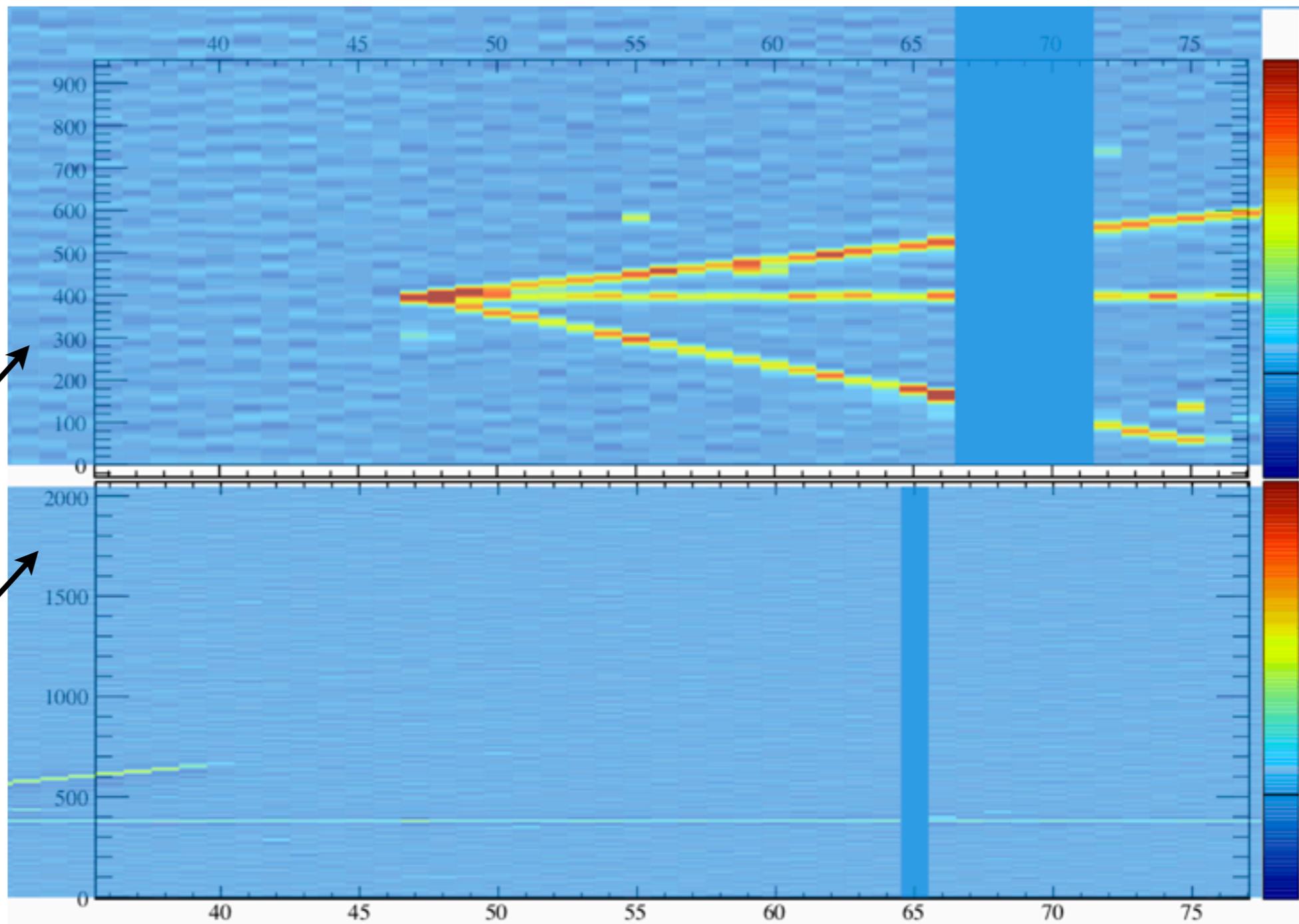
LArSoft
Run: 629/1
Event: 20606
UTC Mon Sep 21, 2009
09:38:12.000000000



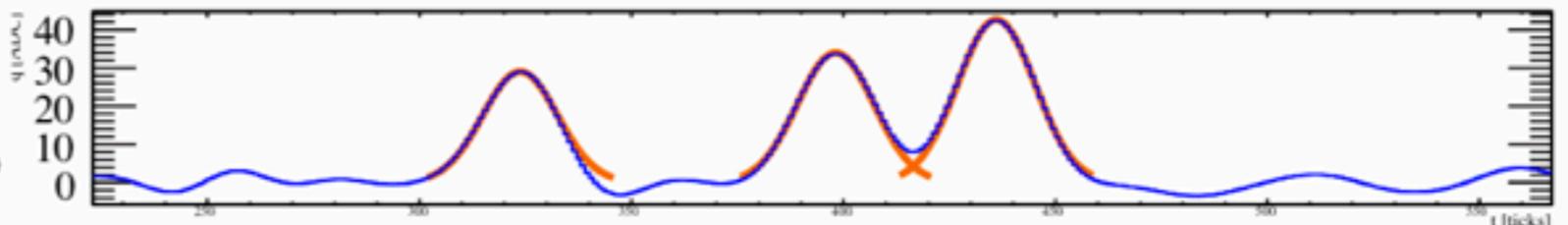
Zooming in on Plane Views

Zooming in on EventDisplay views does not work very well right now.

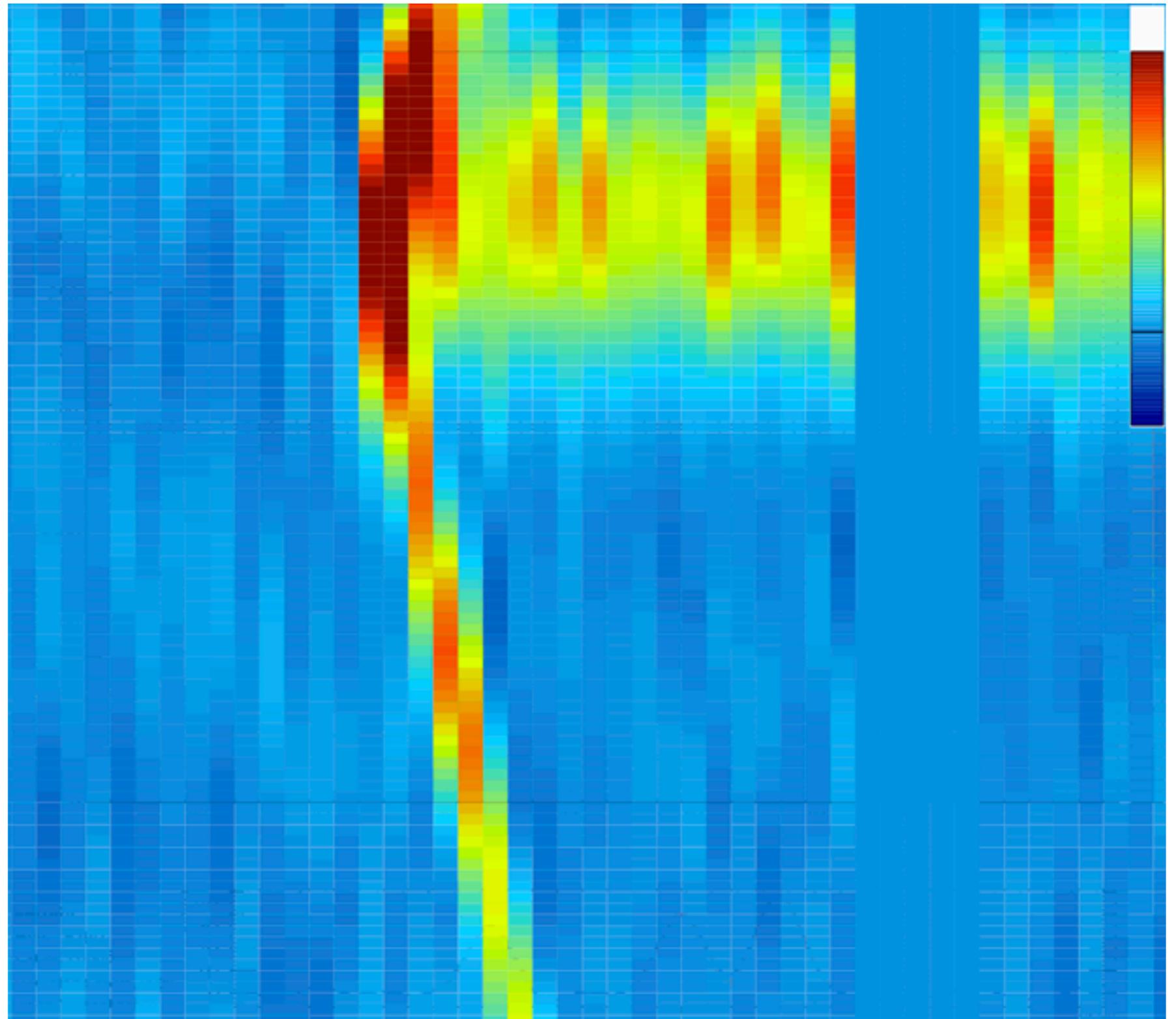
Unlike TH1* objects, TBox objects are not "clipped" to the TFrame that should contain them. (Same behavior is also true for TPolyLines that now represent Hits).



LArSoft
Run: 629/1
Event: 20606
UTC Mon Sep 21, 2009
09:38:12.000000000



Really Zooming In



Just a reminder...the EventDisplay histograms contain information about every single ADC sample for every wire in the TPC.

Some Numbers

- The table below summarizes the main TPC numbers of interest for the EventDisplay.
- “# Samples” is the total number recorded per accelerator trigger (including pre/post sampling)
- ArgoNeuT samples at 5MHz for $410\mu\text{s}$, MicroBooNE at 2MHz for 4.8ms
- A “pixel” is a single ADC sample from a single wire.

	Plane	# Wires	# Samples	# “Pixels”
x2	ArgoNeuT	240	2048	491,520
x2	MicroBooNE Induction	2400	9600	23,040,000
	MicroBooNE Collection	3456	9600	33,177,600

Screen Resolution



- My 15" MacBook Pro has pixel dimensions of 1440x900 (at 110ppi).
- **1,296,000** pixels



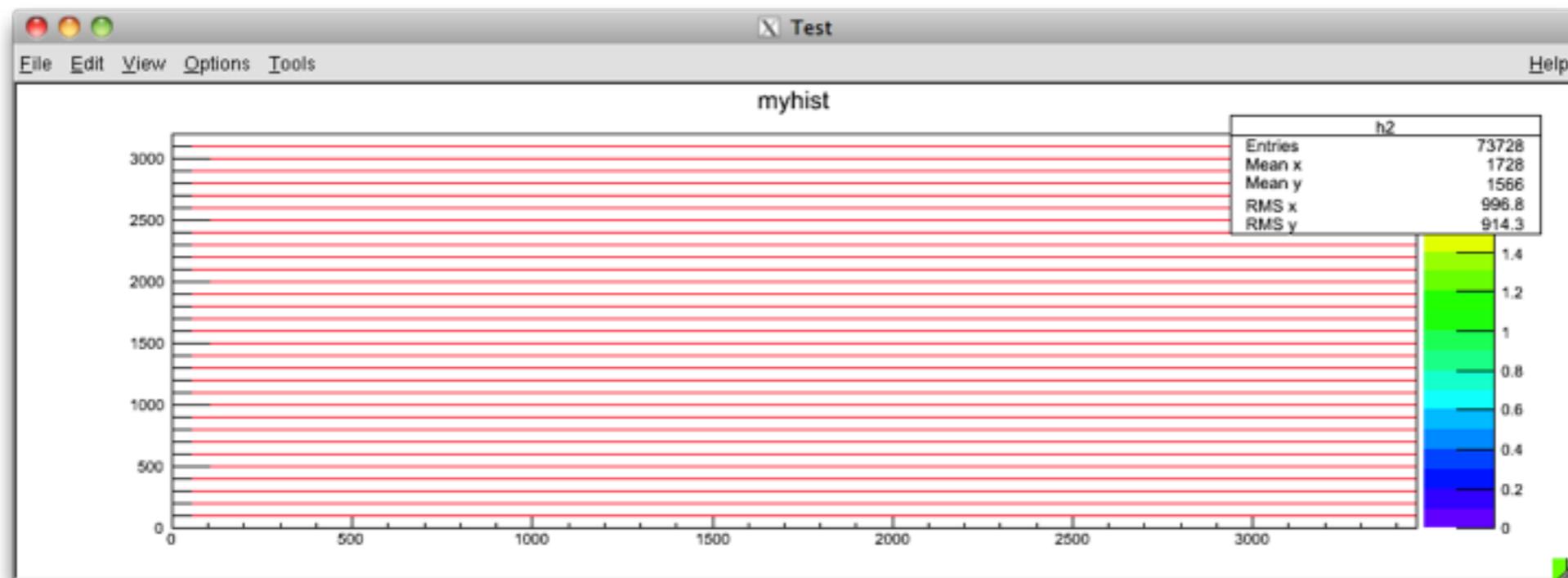
- A typical 30" monitor has pixel dimensions of 2560x1600 (at 101.65ppi).
- **4,096,000** pixels

MicroBooNE events have more “pixels” than your screen...

- ▶ EventDisplay can't simultaneously display all the information it knows about.
- ▶ To simultaneously show all MicroBooNE information for a single plane, with time on the y-axis, we would need to have two side-by-side stacks of 6 30" monitors (which would horribly distort the true aspect-ratio of the events).

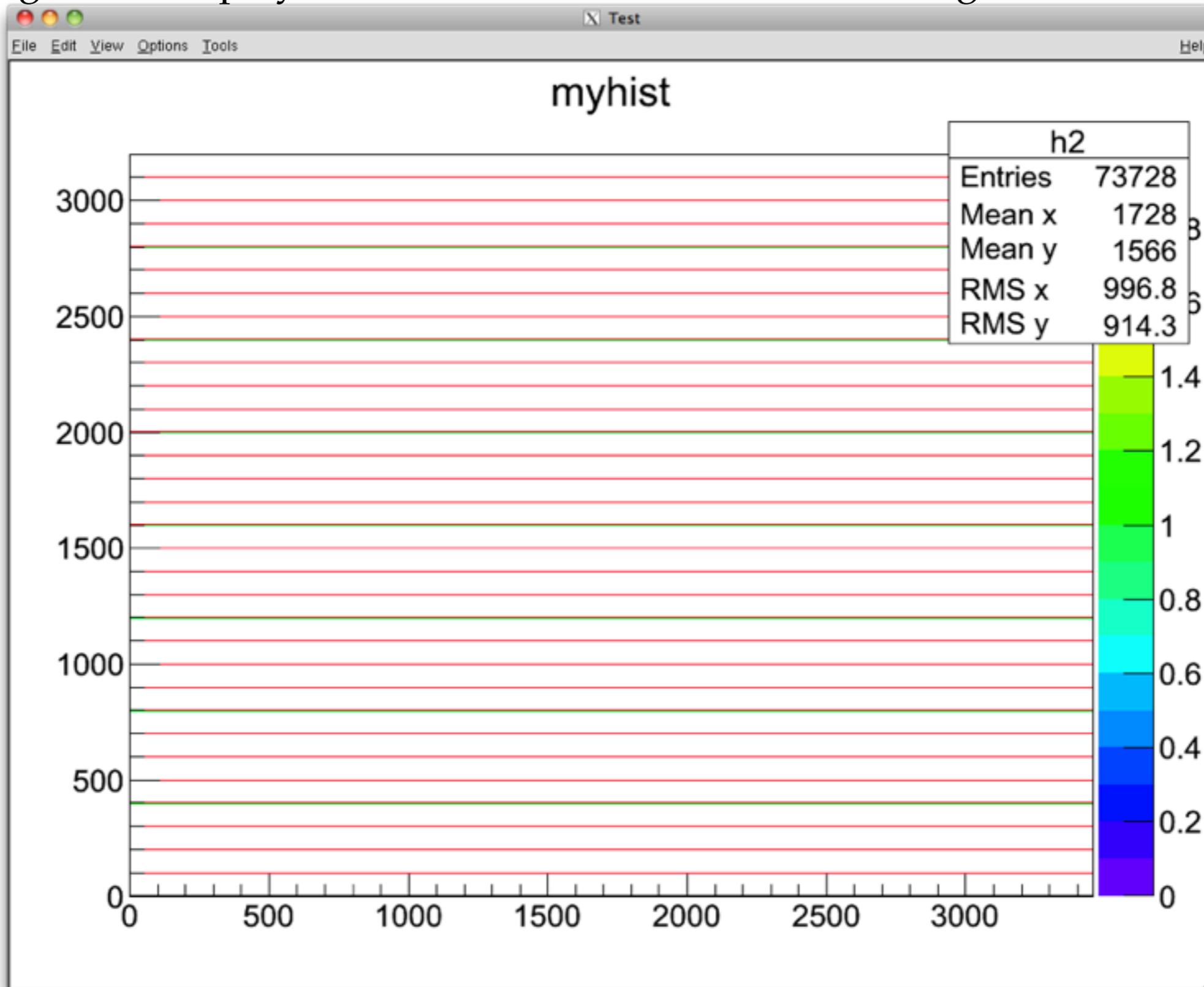
ROOT Behavior

- ROOT is aware of the dimensions of the window it is given to draw histograms within, and will adjust the histogram to fit in that space.
- The image below is a screen-cap of a TH2F histogram I made in ROOT. The histogram is 3200 bins tall x 3456 bins wide (i.e. - the size of one “frame” of the MicroBooNE collection plane, which has 1 / 3 of the 9600 total time samples).
- It appears to be just a bunch of horizontal red lines spaced at a regular $y\%100$ interval.



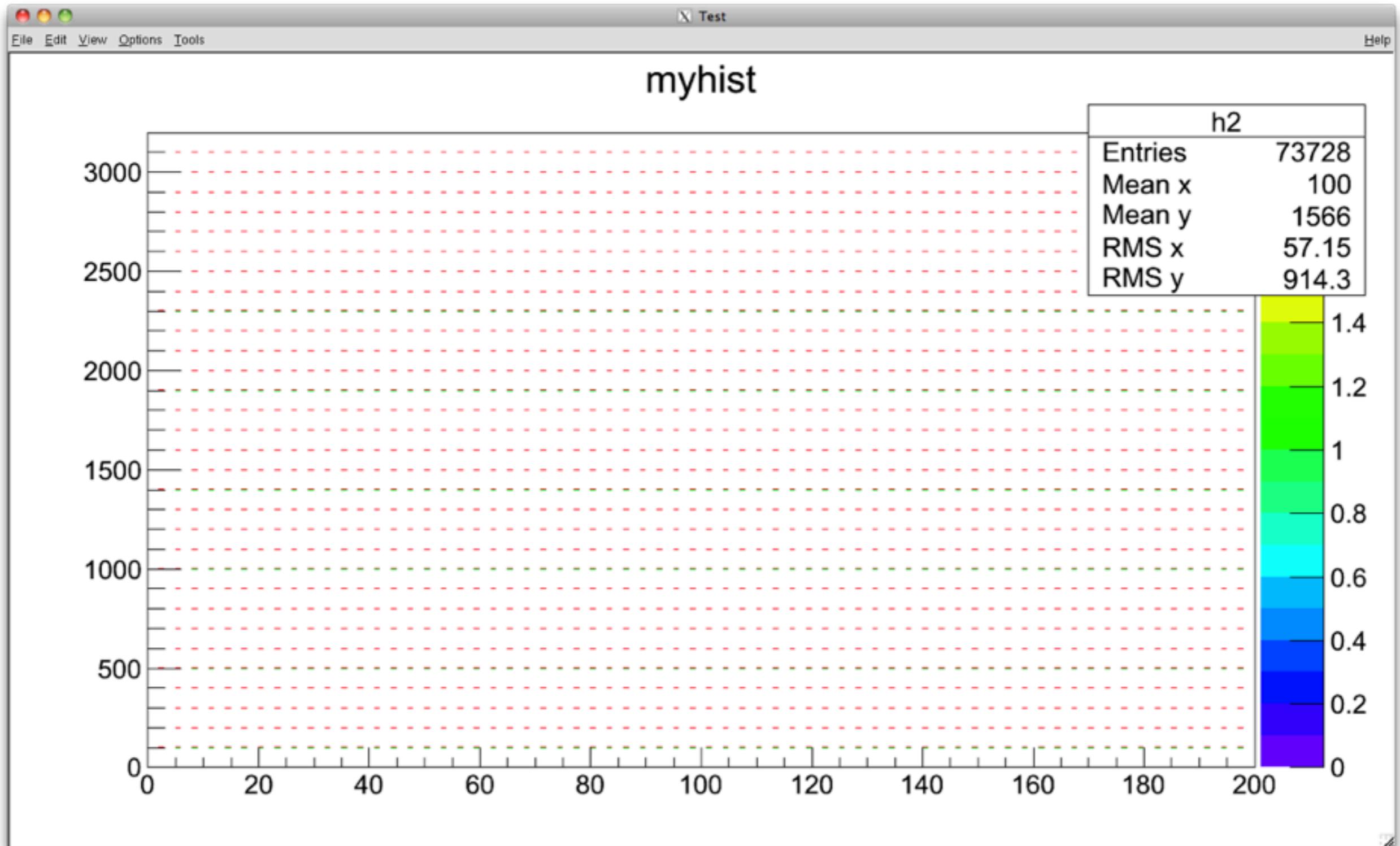
ROOT Behavior

Same histogram is displayed below, but I've increased the "height" of the TCanvas.



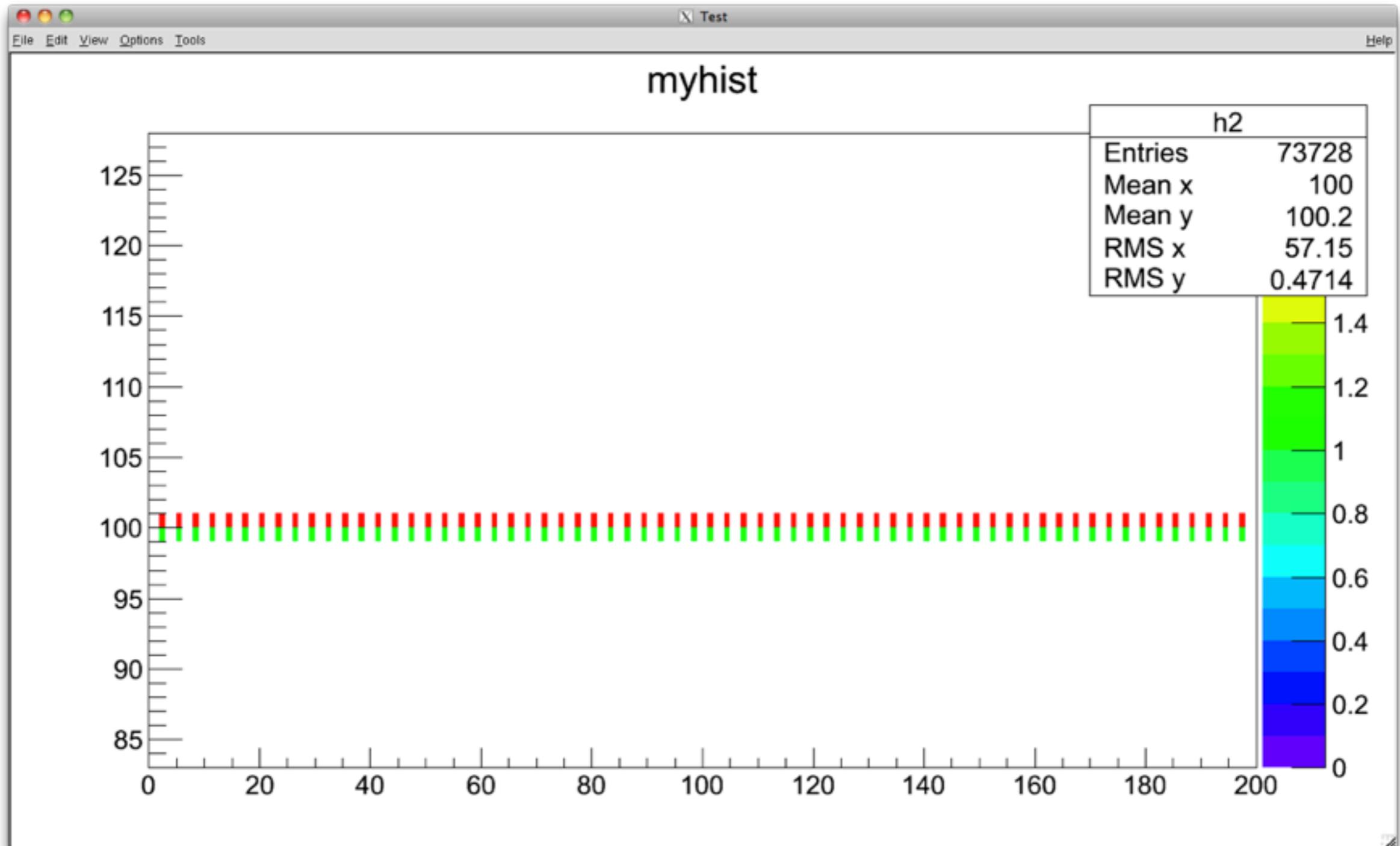
ROOT Behavior

Same histogram, but I've changed the range on the x-axis.



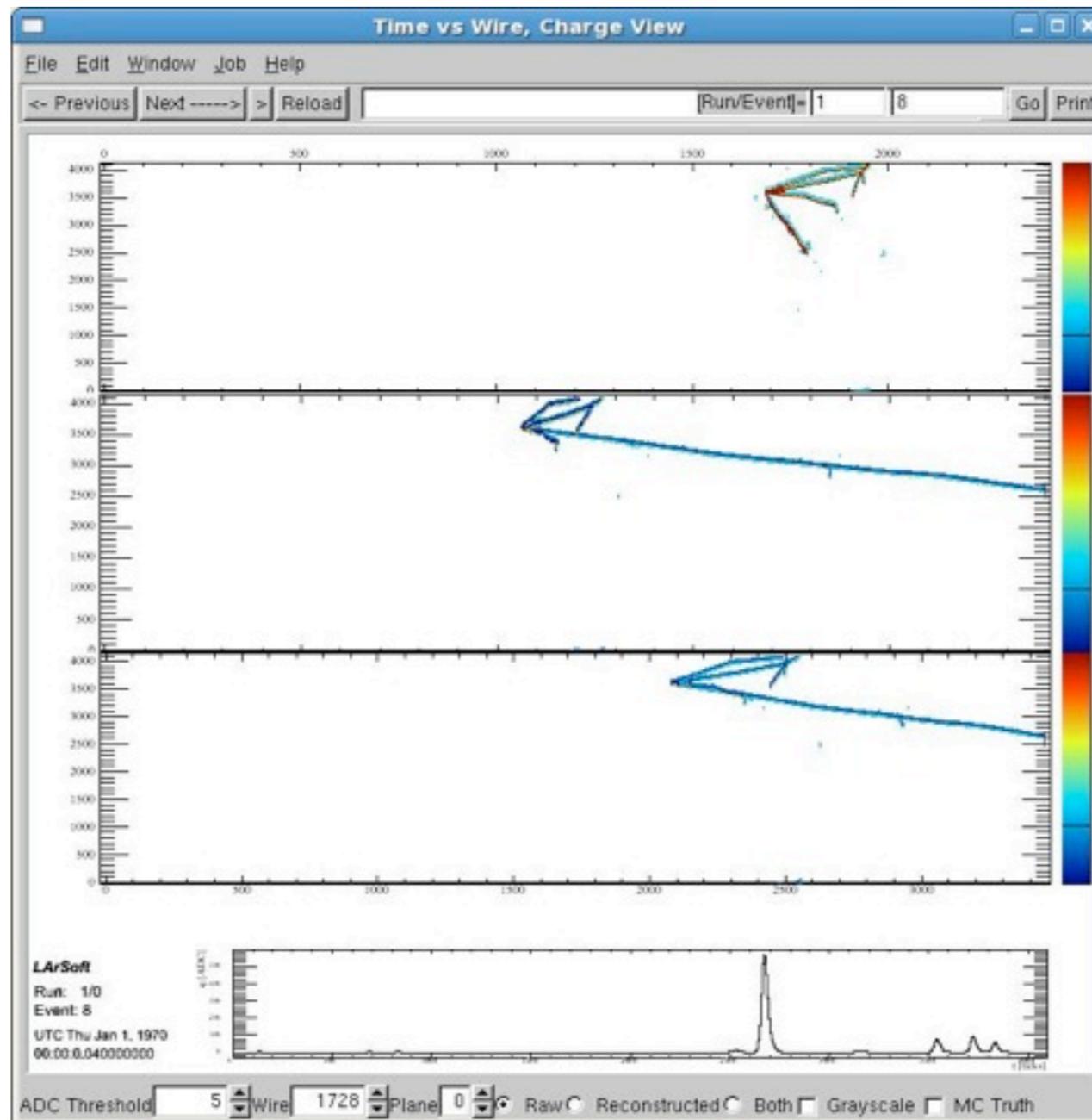
ROOT Behavior

Same histogram, but I've zoomed the range on the y-axis. Now you can see the original histogram had alternating "lines" of red and green. "Lines" were actually discontinuous.



ROOT Behavior

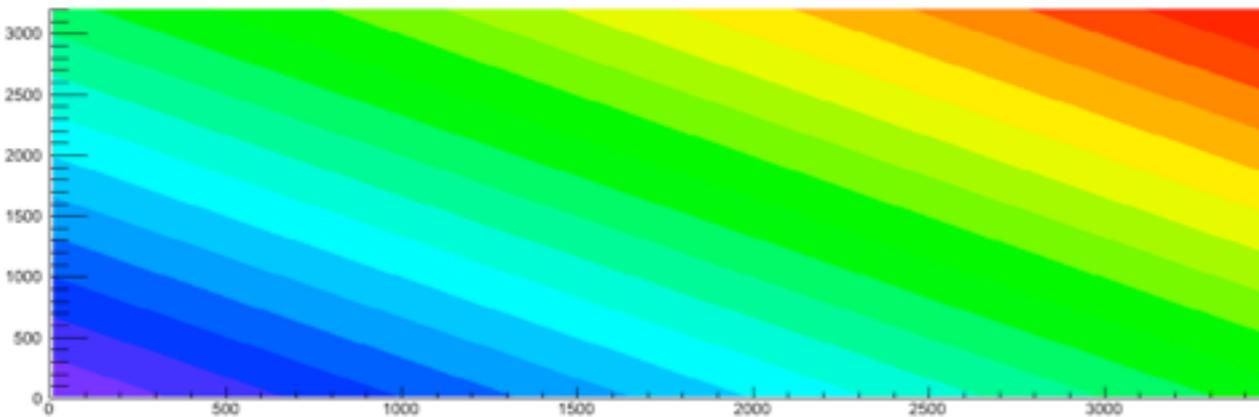
- ROOT automatically rebins histograms to have fewer bins than there are pixels in the allotted drawing space. This rebinning process takes time.
- Drawing all the TPC planes on the same TCanvas necessarily increases the amount of rebinning ROOT will have to do to fit everything in the window.



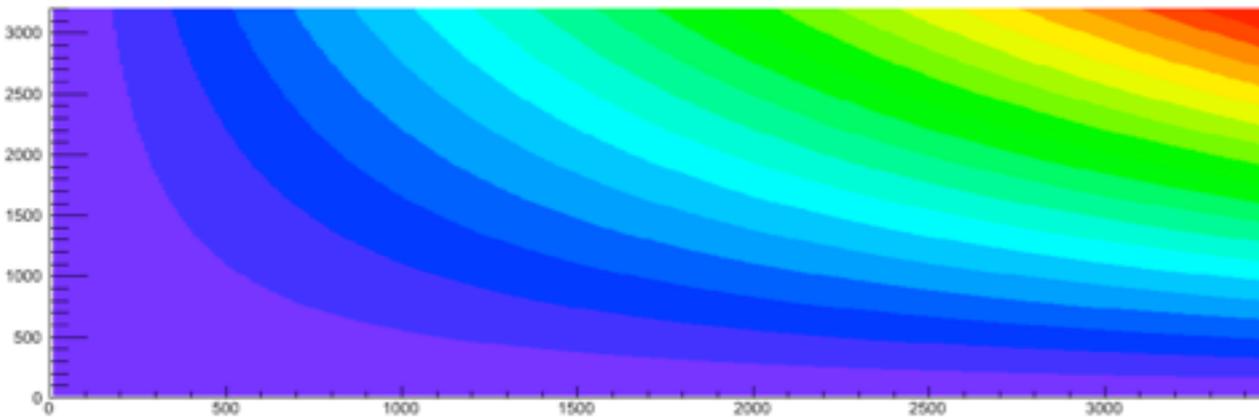
Drawing Speed

- I've started to look back into using TH2F histograms (which don't suffer from the "clipping" problem encountered when zooming) for displaying data.
- First exercise was to draw two histograms in a canvas. In one case (high-resolution), the histograms were 3456x3200, while in the other (low-resolution) they were 346x320.
- The "low-resolution" histograms draw 100 times faster (0.15s) than the "high-resolution" version (~14.2s). **Note:** this is just time to draw, not time to fill histogram.
- Have tried to use multithreading (via ROOT TThread class) to send each histogram to a distinct CPU core...mixed results so far.

myhist

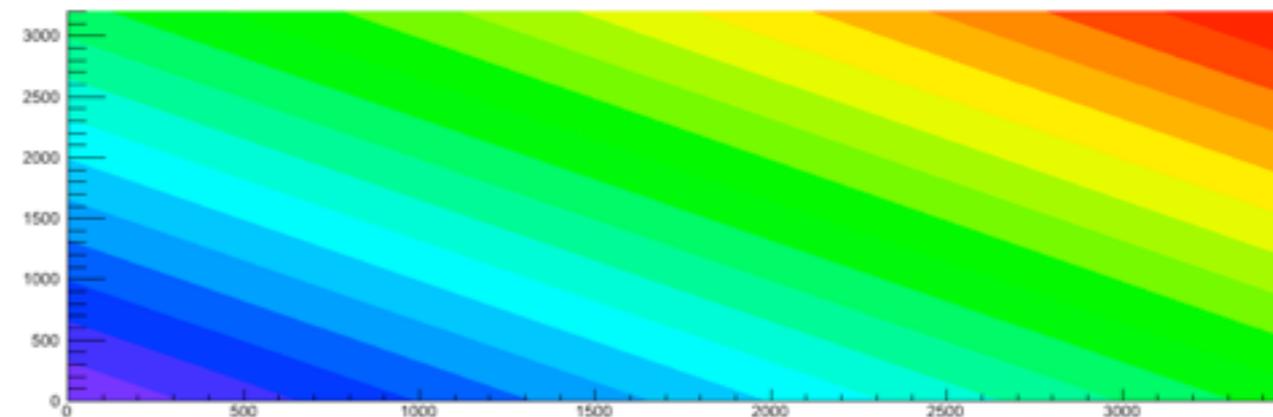


myhist

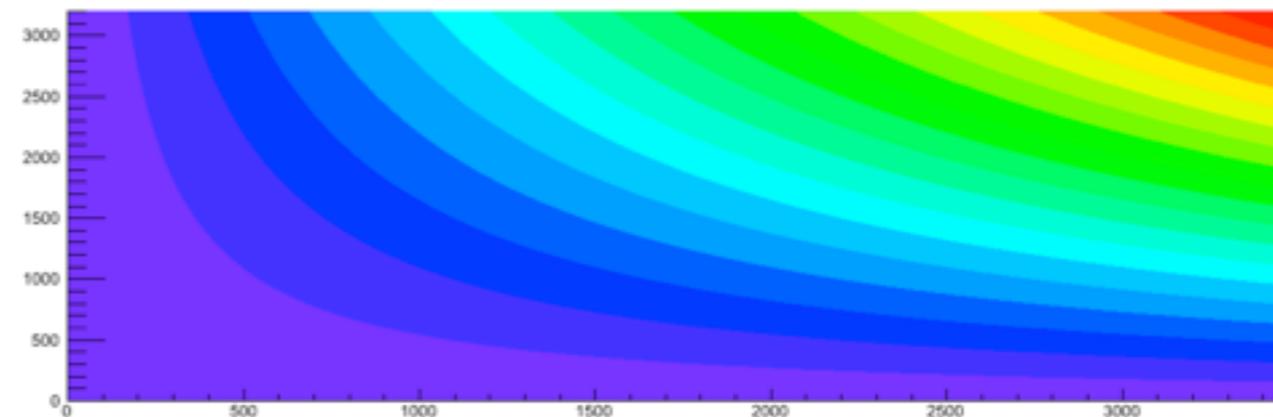


Low-Resolution

myhist



myhist



High-Resolution

Conclusions

- Added new Gaussian shapes for displaying Hits over wires.
- Be aware of ROOTs default rebinning behavior when drawing histograms with more bins than there are pixels on your screen.
 - ▶ Good thing about keeping track of every pixel is it allows zooming in .pdf/ROOT.
 - ▶ Bad thing is the increased memory / CPU required to keep track.
- We need to figure out how to display MicroBooNE data in a reasonable amount of time, which is challenging due to the vast amount of data to deal with.
 - ▶ Low-resolution images for monitoring purposes?
 - ▶ Multi-threading to share the workload and improve speed.
 - ▶ Only draw the Hit objects (i.e. - don't even try to draw the raw / calibrated information unless specifically requested)?
 - ▶ Other ideas...