LMN Resampling for DUNE TPC Raw Data

Brett Viren with Lino Gerlach, Wenqiang Gu, Nitish Nayak, and Xin Qian

March 15, 2024

Topics

- DUNE FD TPC ADC samplings and problems they pose to offline software.
- **Proposal** for mitigation via **resampling** of TPC ADC waveforms.
- Resampling techniques and “LMN” optimization trick.
- Understanding normalization when resampling.
- Example of resampling applied to DUNE FD simulation.

Some LArTPC ADC samplings

Detector	T (ns)	F (MHz)	ADC (bits)
past (PD-1,MB)	500	2.000000	12
DUNE FD			
VD TDE	500	2.000000	14
VD BDE	512	1.953125	14
HD	512	1.953125	14

Sampling period T (“tick”), frequency F and ADC resolution.

A variety of sample periods leads to problems

Some facts.

- DUNE official requirements state softly, "$T \approx 500\,\mathrm{ns}$".
- The mix of $500\,\mathrm{ns}$ and $512\,\mathrm{ns}$ was accepted by DUNE project management.

Headaches for **offline software**.

- VD BD/TD will need two field responses calculations, despite same CRP design.
- Many, otherwise identical configurations require bifurcation, error prone.

Not to be discounted: mental math in easier in $2\,\mathrm{MHz}$ than $1.953\,125\,\mathrm{MHz}$!

Proposal

*Inside the **first processing stage of the Offline Software** (Wire-Cell noise-filter + signal processing) we will **upsample** from $T_s = 512\,\mathrm{ns}$ to $T_r = 500\,\mathrm{ns}$.*

Questions on the proposal

Why standardize on $500\,\mathrm{ns}$?

- Upsampling is **information-lossless** from a sampling-theory perspective.

But, then you must resample 3/4 of the channels!

- Resampling is **low-cost** via **FFT and the LMN trick**, $\mathcal{O}(1\%)$ NF+SP cost.

Why do it first thing in the offline chain?

- Resample **far-upstream** as possible to hide variance from so majority of software.
- Need only single VD field-response calculation, no configuration bifurcation.
- Caveat: mitigating "non-spectral" pathology (stuck-bits, etc) may require original waveforms. Can place special NF component just prior to resampling.

Conventional resampling techniques $N_s \to N_r$

…are typically performed in two integer-factor stages:

$$\begin{array}{rrl} & \text{integer upsample:} & N_s \to N_u = LN_s \\ + & \text{integer downsample:} & N_u \to N_r = N_u/M \\ \hline = & \text{rational}^* \text{ resample:} & N_s \to N_r = RN_s,\ R \equiv \frac{L}{M} \end{array}$$

$$L, M, N \in \mathbb{N}$$

*An irrational R' must be approximated by some integer ratio $R = \frac{L}{M}$.

Summary of conventional resampling

Next, we give a summary over a 2×2 space of:

- The **upsample** and **downsample** steps.
- Operate in **interval** (time) *vs* **frequency** domains.

Conventional (time) **interval-domain** resampling

Upsampling $N \rightarrow N_u = LN$

1. **Insert** $L - 1$ zero-value samples after each of the original N samples.
2. **Convolve** with a smoothing filter, eg (truncated) $sinc()$.

Downsampling $N \rightarrow N_d = \lfloor N/M \rfloor$

1. **Convolve** with anti-alias filter, eg (truncated) $sinc()$.
2. **Decimate** by retaining 1 out of very M consecutive samples.
3. Ignore any trailing remainder when $N/M \notin \mathbb{N}$.

Conventional **frequency-domain** resampling

Upsampling $N \to N_u = LN$

1. **Transform** (DFT) to frequency-domain sequence.
2. **Insert** $(L-1)N$ zero samples above the Nyquist frequency.
   - Equivalent to a perfect smoothing filter.
3. **Transform** (inverse DFT) to interval-domain sequence.

Downsampling $N \to N_d = N/M$

1. **Transform** (DFT) to frequency-domain sequence.
2. **Remove** $(M-1)N$ samples above desired, resampled Nyquist frequency.
   - Equivalent to a perfect anti-alias filter.
3. **Transform** (inverse DFT) to interval-domain sequence.

Caveat: must have signal size N such that $N/M \in \mathbb{N}$ (no remainder).

Foreshadow: note how up#3 and down#1 cancel!

Issues with conventional resampling

Desired R may lead to ruinously large L and M

$R = L/M = N_r/N_s = T_s/T_r = 512\,\text{ns}/500\,\text{ns} \Rightarrow L = 128,\ M = 125.$

Costly time-domain convolution of low-pass downsample filter

It is $\mathcal{O}(N_s N_r)$ while frequency domain operations cost
$\mathcal{O}(N_s \log N_s) + \mathcal{O}(N_s or N_r) + \mathcal{O}(N_r \log N_r)$.

Frequency-domain downsampling requires $N/M \in \mathbb{N}$

We must have a “lucky size” to do this, else use time-domain’s freedom to have a “remainder”.

Can we make our own luck?

The LMN trick

Unified LMN resampling

1. **Calculate** "lucky" size N_s from original size N and desired ratio $R \ni N_r = \frac{L}{M} N_s \in \mathbb{N}$.
2. **Pad or truncate** interval-domain sequence to size $N \to N_s$.
   - May exploit padding to mitigate ringing artifacts.
3. **Transform** (DFT) to frequency domain.
4. **Apply resampling** in one step:
   - Net-upsample: **insert** $N_r - N_s$ samples above **original** Nyquist frequency.
   - Net-downsample: **remove** $N_s - N_r$ samples above **resampled** Nyquist frequency.
5. **Transform** (inverse DFT) to interval domain.

There are some details to properly treat case where a "Nyquist bin" exists due to $N_s, N_r \in \mathbb{N}_{even}$.

Finding LMN's "lucky" N_s

The resampling will shift the sampling frequency by an amount:

$$\Delta F = \frac{1}{T_r} - \frac{1}{T_s} = \frac{T_s - T_r}{T_s T_r}$$

That shift must cover an integer number δN of frequency samples of total bandwidth:

$$\delta F = \frac{\delta N}{N_s T_s}$$

Equating $\delta F = \Delta F$ and solving for N_s:

$$N_s = \frac{\delta N \cdot T_r}{T_s - T_r} \in \mathbb{N}$$

Finding LMN's minimal δN

Continuing:

$$N_s = \frac{\delta N \cdot T_r}{T_s - T_r} \in \mathbb{N}$$

A (minimum) value of δN that satisfies $N_s \in \mathbb{N}$ is:

$$\delta N = \frac{T_s - T_r}{\gcd(T_r, T_s - T_r)}$$

Where $\gcd(\cdot, \cdot)$ is the real-valued greatest-common-denominator function.

- The **extended Euclidean algorithm** is recursive but fast to converge.
- Any multiple of δN may be chosen, gives freedom to pick N_s closest to a "natural" size.
- LMN rationality: T_s and T_r must be such that $\delta N \in \mathbb{N}$ can be found, ie $\gcd()$ converges.

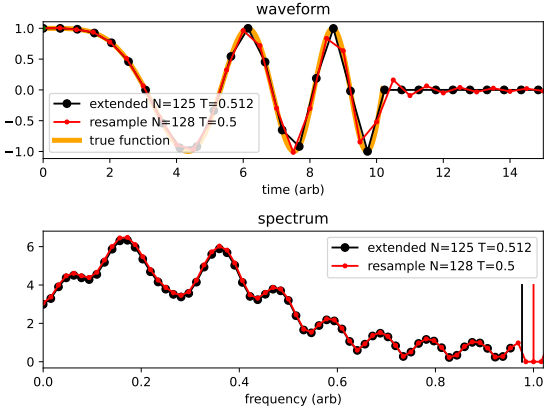
Example 1 - the finite cosine chirp $T_s = 0.512 \rightarrow T_r = 0.500$

Original $N = 20,\ T_s = 0.512$ (arb units)

- Want to resample to $T_r = 0.500$.
- LMN says: $\delta N = 39,\ N_s = 125$.

Note,

- High frequency **ringing**.
  - Tail extension may be truncated away.
  - A later low-pass filter can be used.



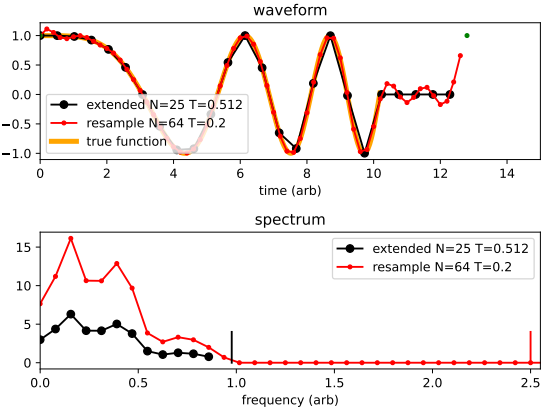
Example 2 - the finite cosine chirp $T_s = 0.512 \rightarrow T_r = 0.200$

Original $N = 20,\ T_s = 0.512$ (arb time)

- Want to resample to $T_r = 0.200$.
- LMN says: $\delta N = 3,\ N_s = 25$.

Note,

- **Periodicity** is violated
  - Tail extension may be truncated away.
  - But still large errors at signal beginning.
- Can be corrected with smarter padding fill.
  - Eg, linear or half-cosine interpolation.
  - Examples in backup slides.



Normalizing when resampling $N_s \to N_r$

LMN uses a DFT round trip which gains overall normalization of $\frac{1}{N}$.

- Dominant software convention: apply $\frac{1}{N}$ to inverse DFT.
- When resampling, normalization is $\frac{1}{N_r}$ (not $\frac{1}{N_s}$).

Must also respect a **physical normalization** that depends on the **signal interpretation**.

Interpolation: interval-domain signal **magnitude** is conserved.

Must **replace** conventional $\frac{1}{N_r}$ normalization with $\frac{1}{N_s}$, multiply resampled signal by $\frac{N_r}{N_s}$.

Integral: interval-domain signal **integral** is conserved.

The conventional $\frac{1}{N_r}$ is correct.

Energy: samples represent field/probability "amplitude", conserve **Parseval energy**

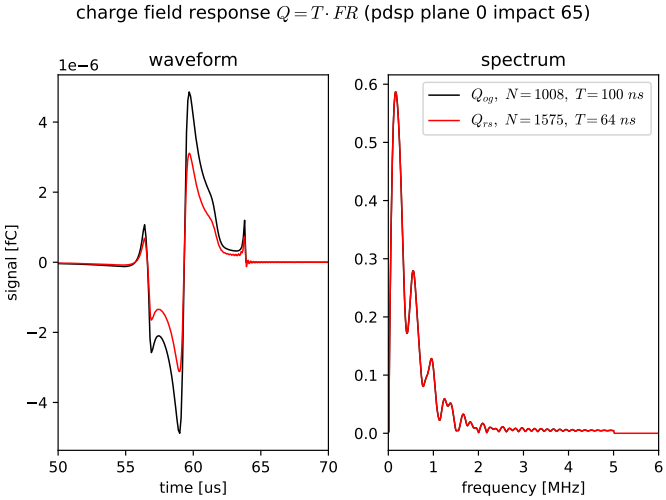
Must multiply resampled sequence by $\sqrt{\frac{N_r}{N_s}}$.

Example 3 - LArTPC simulation

Strategy for this illustration:

- Apply LMN to resample field response: $T_s = 100\,\text{ns} \rightarrow T_r = 64\,\text{ns}$.
  - Saves us from having to rerun GARFIELD.
  - Itself, provides illustration of LMN.
- Run Wire-Cell sim at $500\,\text{ns} = 5 \times 100\,\text{ns}$ and $512\,\text{ns} = 8 \times 64\,\text{ns}$.
- Resample ADC ticks: $T_s = 512\,\text{ns} \rightarrow T_r = 500\,\text{ns}$
- Compare resampled $500\,\text{ns}$ to native $500\,\text{ns}$.

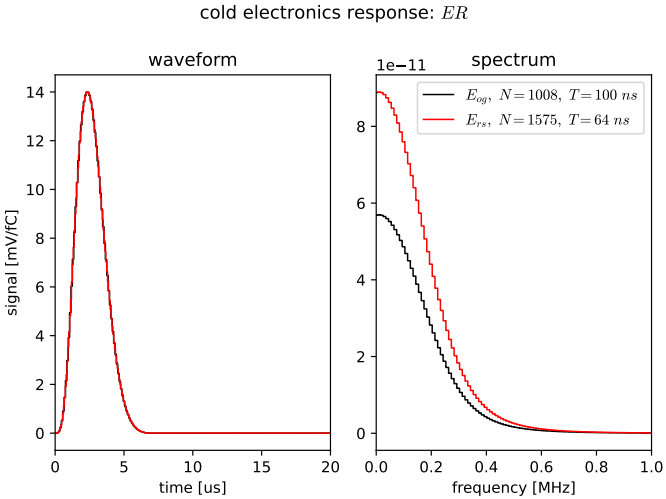
LMN-upsampling of charge field response 100 ns → 64 ns



This is **charge** (not current):

- must **interpret as integral samples**: conserve signal sum.

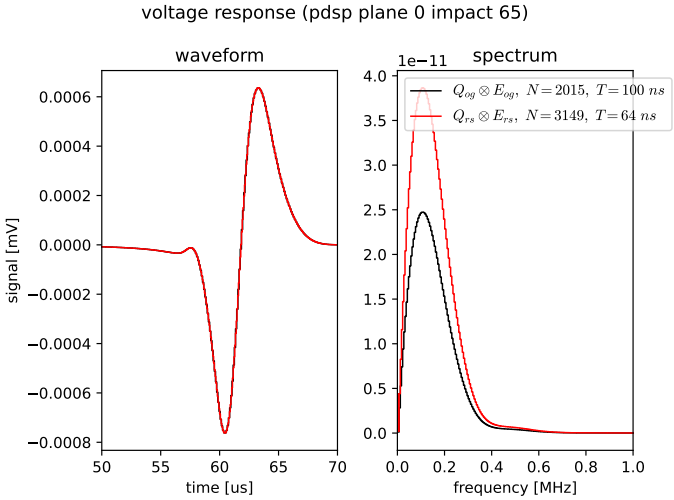
Electronics response at 100 ns and 64 ns



ER interpretation: interpolation

- Actually, not a **resampling** but simply a sampling of an analytic function at 100 ns and 64 ns.

Detector voltage response ($Q \otimes ER$) at 100 ns and 64 ns

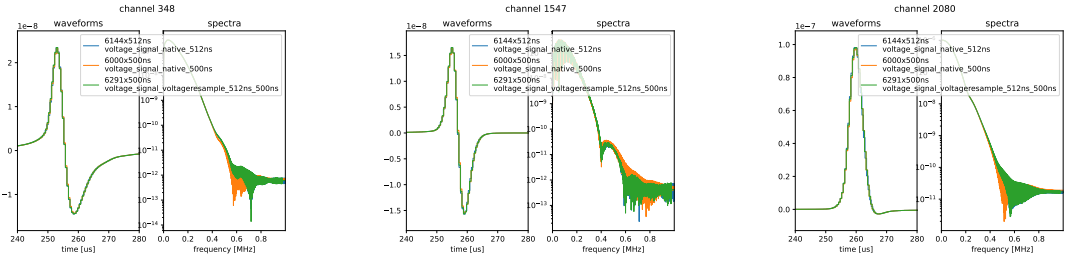


Convolution $T \cdot FR \otimes ER$ at the two samplings.

- Expect and get **perfect** effective interpolation of voltage sampling.

Foreshadow: plot is from an external calculation, not from WCT sim.

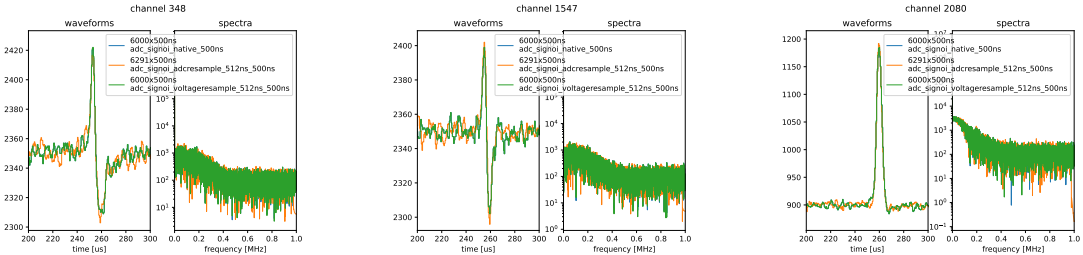
WCT sim resampling - voltage (signal only)



One select channel from each view recording an ideal track going diagonal across detector.

- Some apparent distortion between native and resampled.
- This is **not expected** and if found to be caused by a "feature" of WCT $FR \otimes ER$.
  - Slides in backups that explains the details.

WCT sim resampling - ADC (signal+noise)



Same track and channels in ADC with resampling at voltage level or at ADC level.

- Noise is common between blue (native) and green (resample at voltage).
- Orange (resample at ADC) is representative of real-world.
- With noise and digitization, the distortion due to bad $FR \otimes ER$ is not significant.
  - Still, we will fix it.

Paper

“Data Waveform Re-sampling with LMN Method”

L. Gerlach, B. Viren, W. Gu, N. Nayak, and X. Qian

Sections

- LArTPC signal processing,
- LMN resampling method,
- Performance comparison with other methods,
- Application to LArTPC.

Status - all but done…

Will first fix the WCT $FR \otimes ER$ issue in order to make the story simpler.

Summary

Proposed: Wire-Cell will resample DUNE FD TPC to uniform samplings.

- $512\,\mathrm{ns} \rightarrow 500\,\mathrm{ns}$ for HD and VD-BD (VD-TD already at $500\,\mathrm{ns}$)
- Apply just prior to WCT NF+SP, or in NF if non-spectral mitigation needed.

Sparked development of the “LMN trick”.

- Faster than conventional resampling (use FFT, no large intermediate signals).
- Allows for perfect, low-pass filter (important for downsampling).
- Very simple / obvious, but so far we have not found it in literature searches.

Some general “resampling gotchyas” explored.

- Avoid artifacts induced by periodicity violation, “kinks” in padding.
- Proper normalization given a physical interpretation of the signal.

Paper writing close to done.

- Must first fix WCT sim $FR \otimes ER$ for nicest plots.

$\mathcal{FIN}$

Some links

- Working repo area

https://github.com/BNLIF/lmn/

- LMN functionality migrating to `wirecell.util.lmn`.

https://github.com/wirecell/wire-cell-python

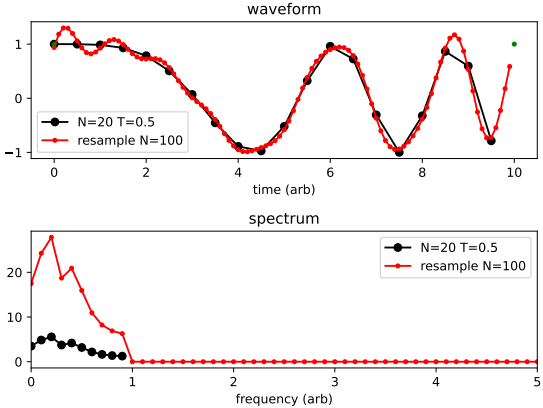
- Paper in overleaf

https://www.overleaf.com/project/65525feda20cdec448ee4390

Non-periodic ringing example

Lack of **smooth wrap-around connection** leads to “wiggles” near endpoints.

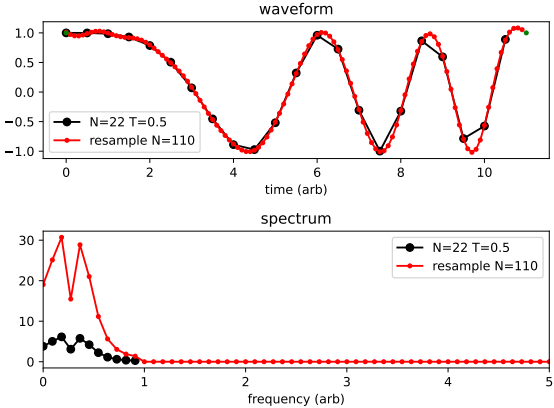
- $R = 5$ net-upsample from black to red.
- Red upsample has oscillatory artifacts.
- Green dot marks initial value at end.



Constructing periodicity with true function

Simply continue to sample original function.

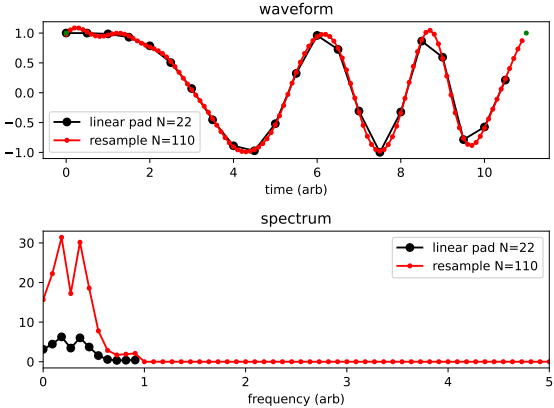
- Extrapolation is perfect as we know the underlying function.
- But, not perfectly smooth so some artifacts remain.
- Of course, function is unknown in measured waveforms.



Constructing periodicity with linear extension

Linearly extend original waveform.

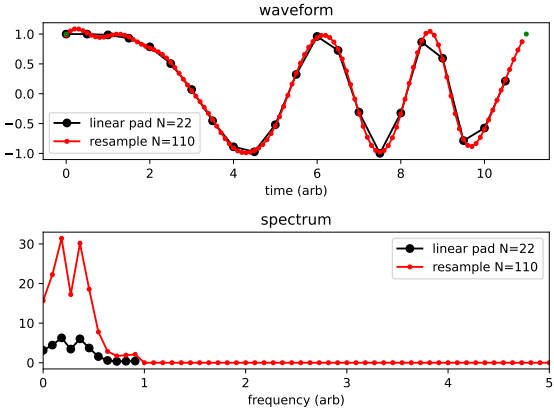
- Kinks can induce high-frequency power.
  - This power may become lost in net-downsampling or induce high-frequency ringing in net-upsampling.



Constructing periodicity with half-cosine

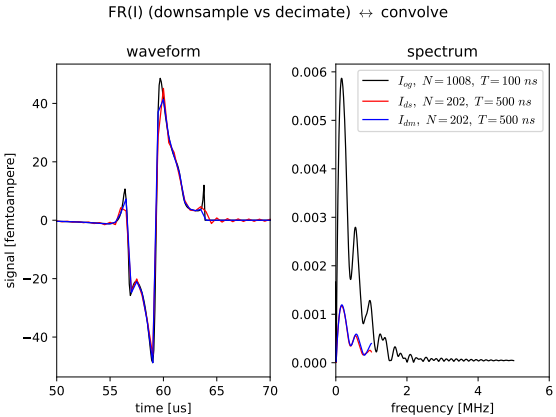
Extend original waveform with half-cosine.

- Care is needed to keep cosine frequency inside original signal bandwidth.
  - O.w. same high-frequency loss/ringing as with linear kink.
- Note, this example is identical to linear as we extend by a single green point.

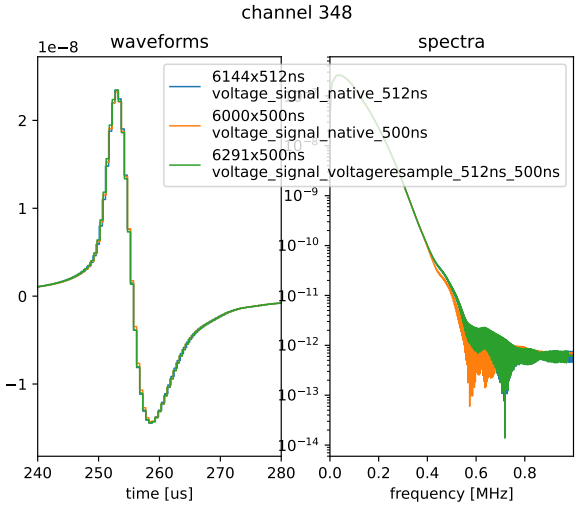


Response - (downsample, decimate) ↔ convolve

- FR at 100 ns has power above the Nyquist frequency for 500 ns sampling (1 MHz).
- Decimation will **alias** this high-frequency power down to lower frequencies.
- Downsampling will **filter** it away.



But, what is that distortion?!



Unexpected differences in native $512\,\mathrm{ns}$ (using resampled FR), native $500\,\mathrm{ns}$ and resampled $512\,\mathrm{ns} \rightarrow 500\,\mathrm{ns}$.

- Not due to HF ringing from undersampled FRs.
- Not due to periodicity violation.
- Not seen in external $T \cdot FR \otimes ER$ plot just shown.

Likely arising in WCT's $FR \otimes ER$ calculation.

WCT's $FR \otimes ER$

- FR is a "fast" signal, nominally sampled at $100\,\mathrm{ns}$ (eg by GARFIELD).
- ADC is a "slow" signal at $500\,\mathrm{ns}$
- WCT sim must, at some point, downsample $100\,\mathrm{ns} \to 500\,\mathrm{ns}$.

2×2 set of choices in how to do that as part of $FR \otimes ER$:

- **convolution-then-downsample** *vs* **downsample-then-convolution**.
- downsample with **simple decimation** or **downsample with filter**.

Turns out, three of these are equivalent while WCT implements the one deviant.

- **decimate-then-convolve**

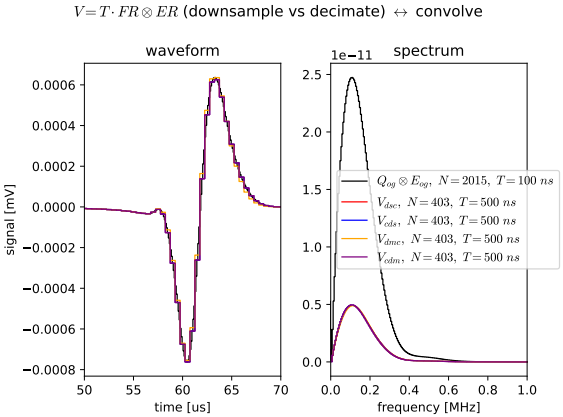
Check 2×2 choices - (downsample, decimate) $\leftrightarrow$ convolve

Look at all 2×2 choices $(ds, dm) \leftrightarrow c$.

- decimate-then-convolve (dmc) is deviant, the other three are coincident.
- decimate-then-convolve worse representation of original 100 ns.

Lack of anti-alias, low-pass filter in **decimation** leads to HF aliasing (feed down).

- This is a **classic sampling mistake!** Doh!



Again, not significant once noise is added but we will fix this.

Why 3 of the 2×2 are equivalent?

Why $dsc \approx cds$?

The **downsample** and **convolve** operations are order independent.

- Ignoring whatever is happening in the low power, high frequency tail…

But, why?

- Interval domain convolution $a \otimes b$ is frequency domain multiplication $A \times B$.
- Resampling $x_s \to x_r$ in LMN style is padding/truncation in frequency domain $X_s \to X_r$.
  - This is effectively applying a “step” aka “box” filter $\to$ frequency domain multiplication!
- Multiplication is commutative.

Why $cdm \approx cds$?

Similar reason to “Why $dsc \approx cds$?” after recognizing that the ER in the **convolution** is a good **low-pass anti-alias filter** so the **decimation** does not cause aliasing.

DFT time-shift theorem

A **constant shift in interval domain** is a **linear phase shift in frequency domain**.

$$x_n \leftrightarrow X_k, \Rightarrow x_{n-l} \leftrightarrow X_k e^{-i2\pi kl/N}$$

To search for shifts between signals a_n and b_n, may plot:

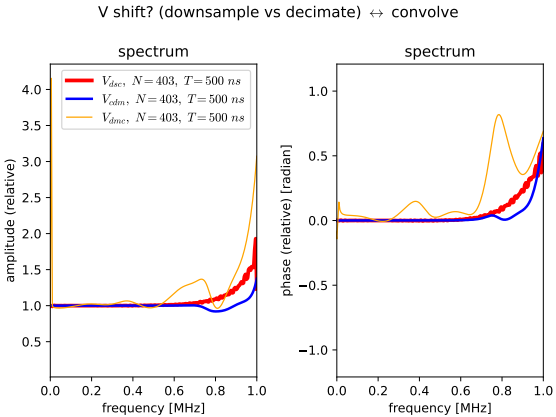
relative magnitude $|A_k|/|B_k|$

relative phase $phase(A_k) - phase(B_k)$.

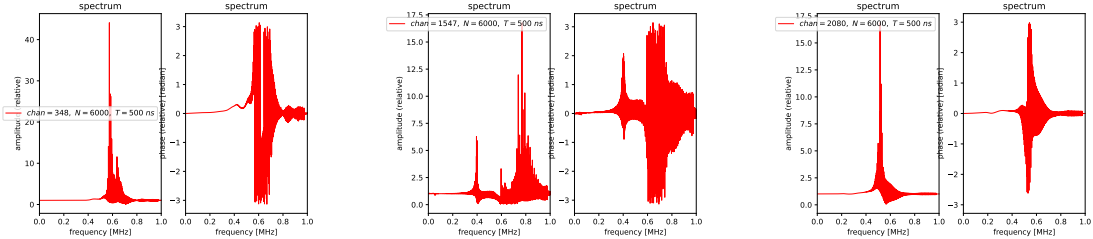
Shift check for $T \cdot FR \otimes EF$

No over-all time shift is seen, but:

- All choices produce deviations in power and time shift at high-frequency, where there happens to be little to no power.
- Decimate-then-convolve (dmc) results in "frequency-local" time shifts where there is actually some signal power.
  - This is likely aliasing in action.



Shift check for WCT voltage



This shows same "shift check" plot for full WCT voltage convolution, not just single FR path.

- To complex to get much insight other than it being the superposition of many paths.
- When fix $FR \otimes ER$ to be **convolve-then-downsample** expect these plots to be flat.