DEEP CLEAN

RICH ORMISTON

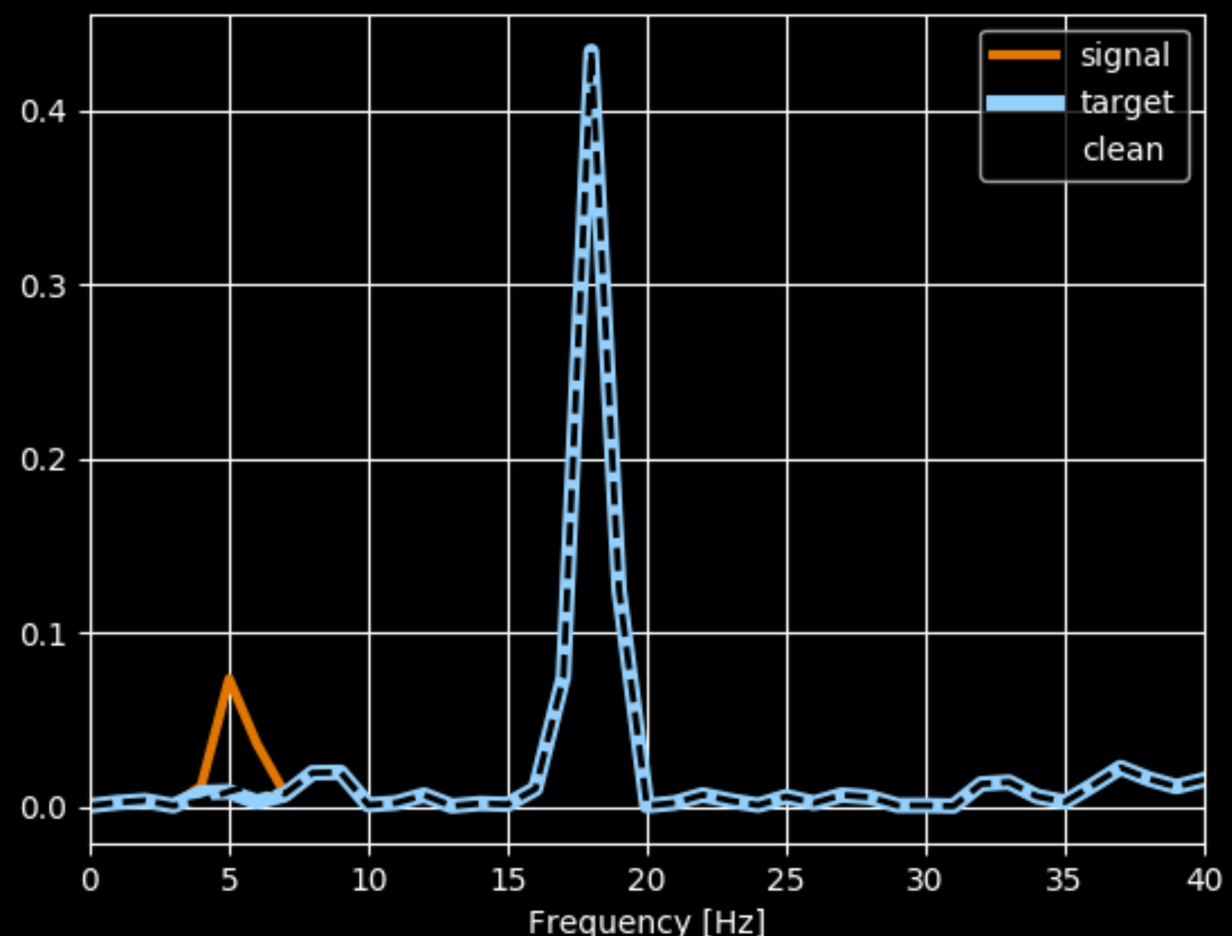
DEEPCLEAN GROUP MEETING 11/08/2019

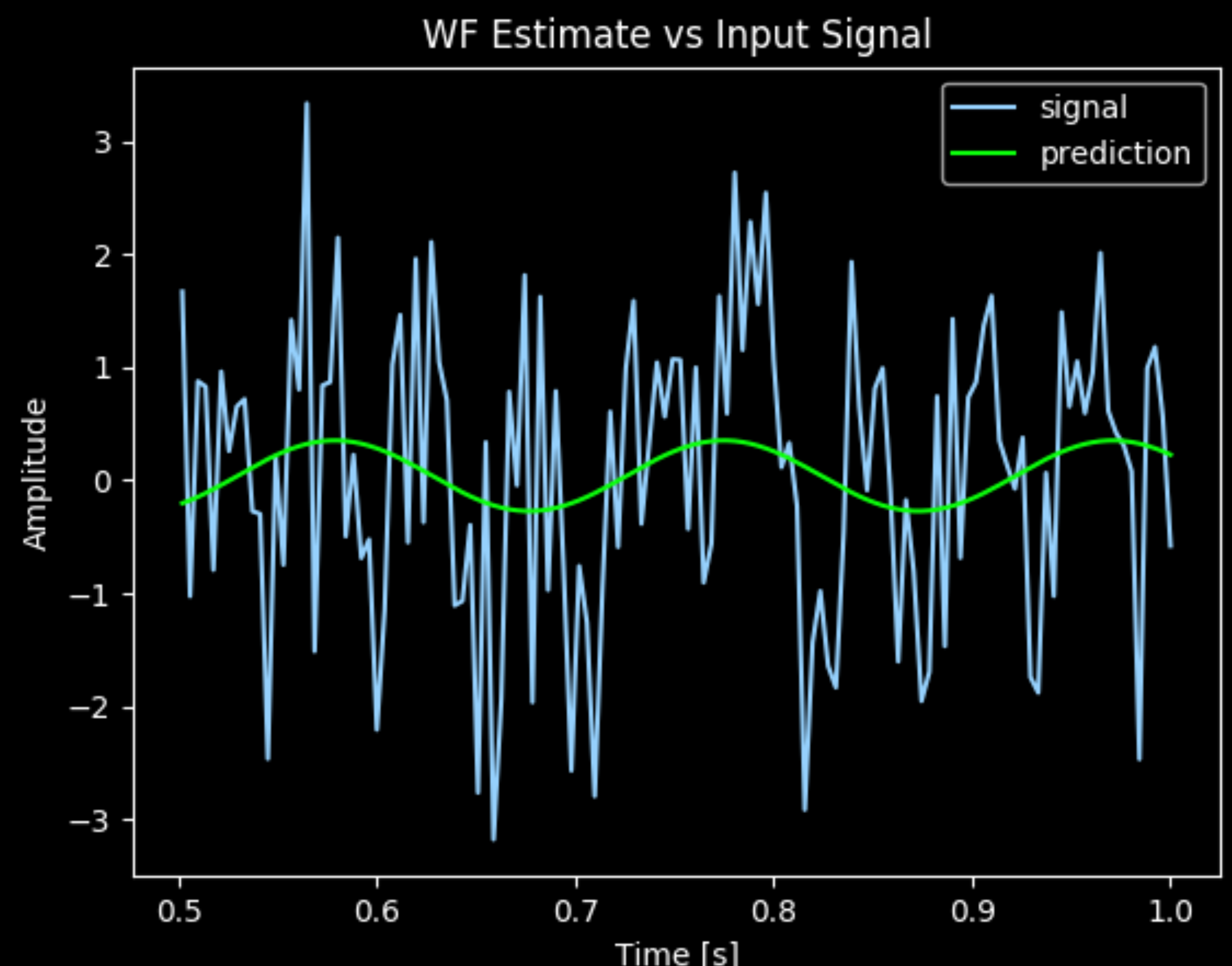
FILTER USE HISTORY

- Wiener filters
  - linear, heavily used
- Extended Wiener filters
  - linear, bicoherent, not used
- Adaptive filters
  - linear, non-stationary, not used (unsure of attempts to apply)
- Extended adaptive filters
  - linear, bicoherent, non-stationary, Gabriele's code is similar
- 2nd order volterra filter
  - nonlinear, non-stationary, not used (or tried as far as I know)

FILTERING: WF IN TIME DOMAIN

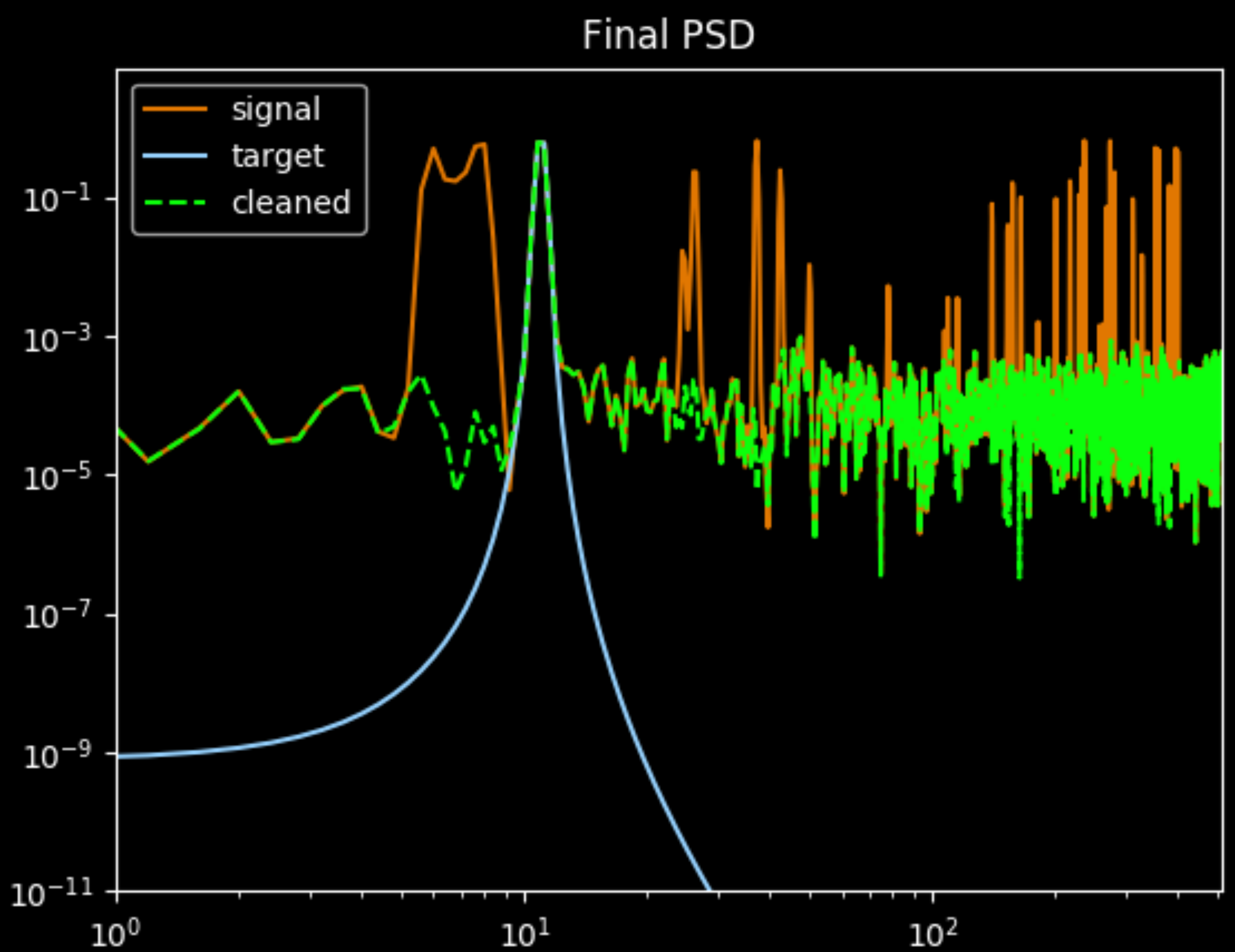
$$\frac{\partial \left\langle e^2[k] \right\rangle}{\partial a_j} = 0 = \frac{\partial}{\partial a_j}\left(d[k] - \sum_{n=0}^{N-1} a_n w[k-n] \right)^2$$

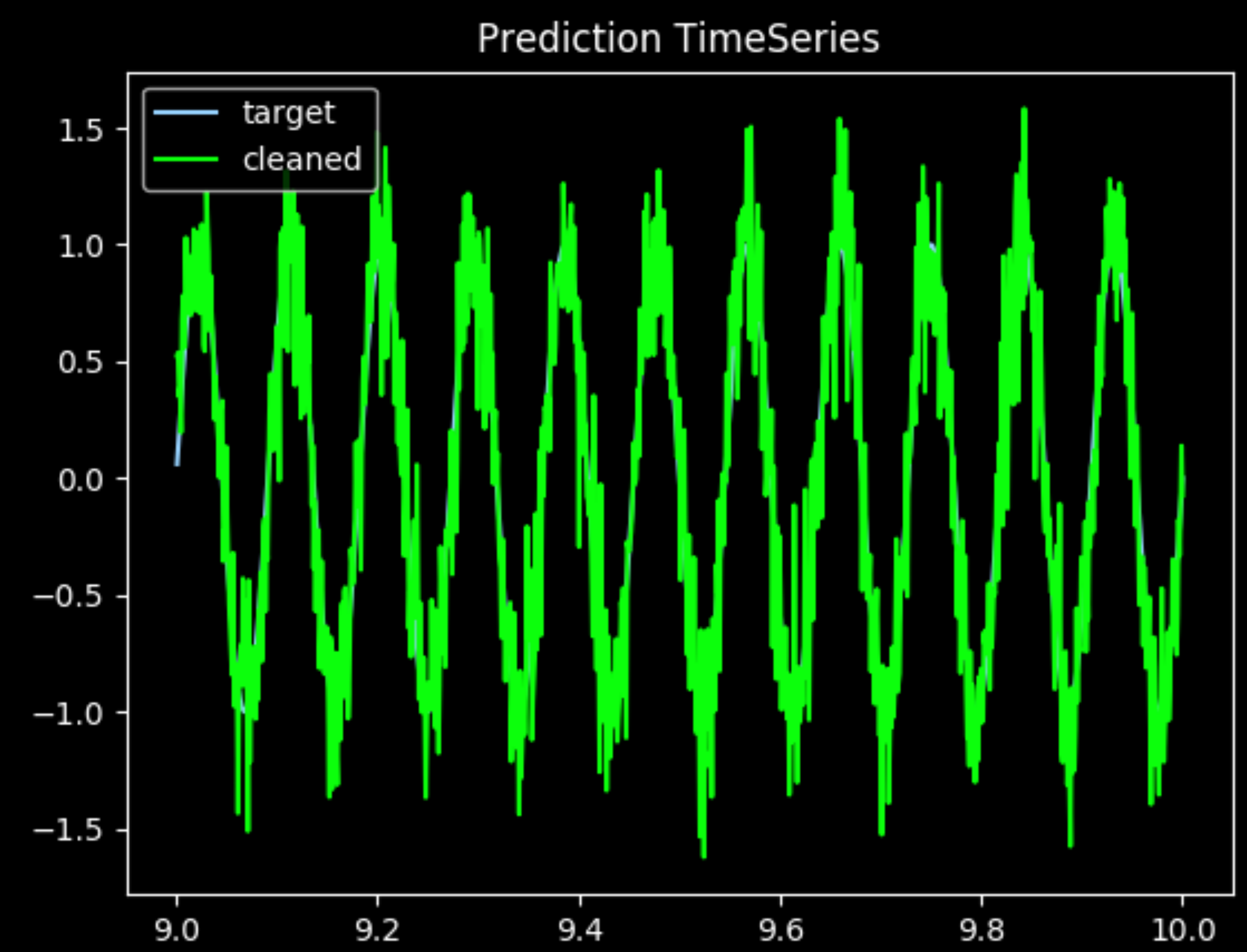




FILTERING: WF IN TIME DOMAIN

$$\frac{\partial \left\langle e^2[k] \right\rangle}{\partial a_j} = 0 = \frac{\partial}{\partial a_j} \left(d[k] - \sum_{n=0}^{N-1} a_n w[k-n] \right)^2$$

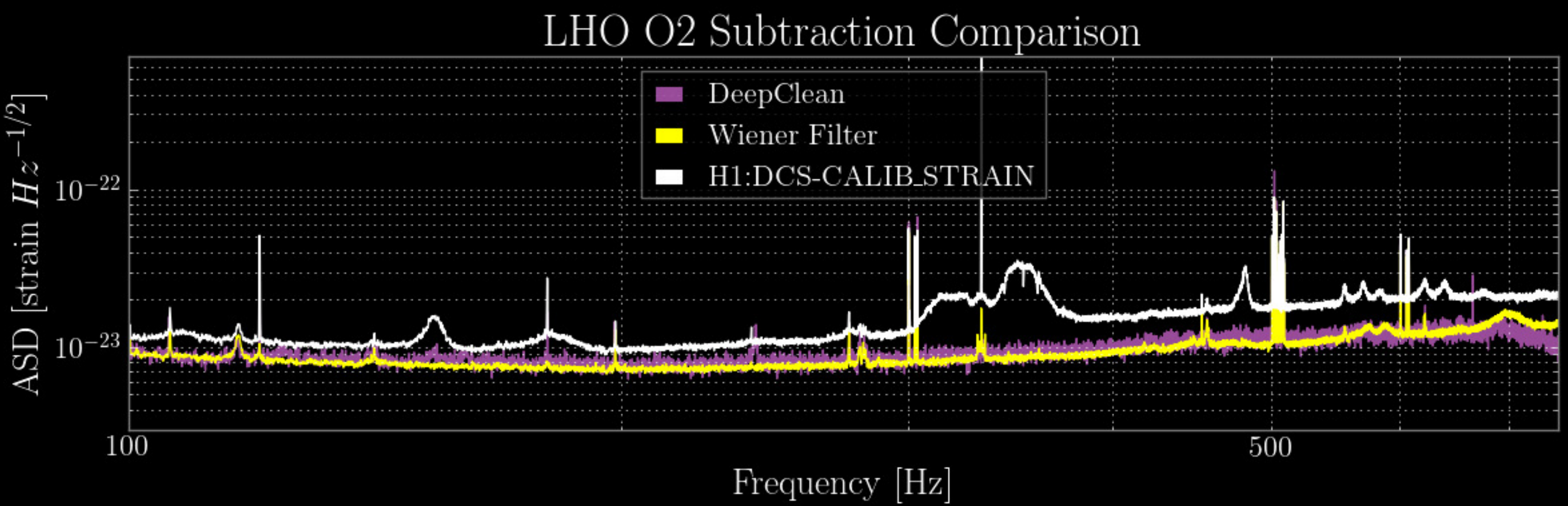




MLAs do linear subtraction well too

FILTERING: WF IN TIME DOMAIN

$$\frac{\partial \left\langle e^2[k] \right\rangle}{\partial a_j} = 0 = \frac{\partial}{\partial a_j} \left(d[k] - \sum_{n=0}^{N-1} a_n w[k-n] \right)^2$$



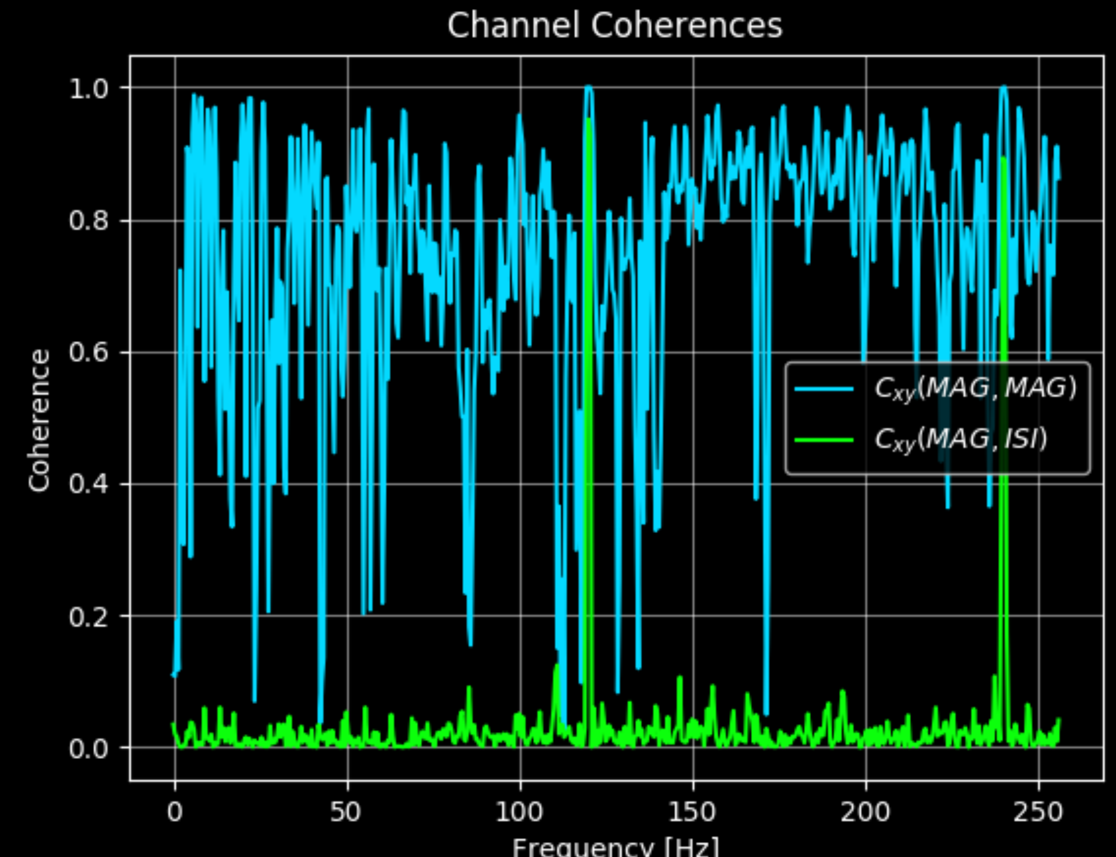
MLAs do linear subtraction well too

FILTERING: MULTI-CHANNEL WF

$$\frac{\partial \left\langle e^2[k] \right\rangle}{\partial a_j} = 0 = \frac{\partial}{\partial a_j} \left(d[k] - \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} a_n^{(m)} w^{(m)}[k-n] \right)^2$$

$$\longrightarrow \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} a_n^{(m)} R_{ww}^{mp}[k-n] = R_{ws}^{p}[k]$$

- M auxiliary channels and a max lag of N steps
- Possible that not all cross terms are zero (i.e, PEMs are not necessarily orthogonal)



REGRESSION OF ENVIRONMENTAL NOISE IN LIGO DATA (LIGO-P1400004-V4)

In the presented approach the Wiener-Kolmogorov method has been extended, incorporating banks of Wiener filters in the time-frequency domain, multi-channel analysis and regulation schemes, which greatly enhance the versatility of the regression analysis. Also we presents the first results on regression of the bi-coherent noise in the LIGO data.

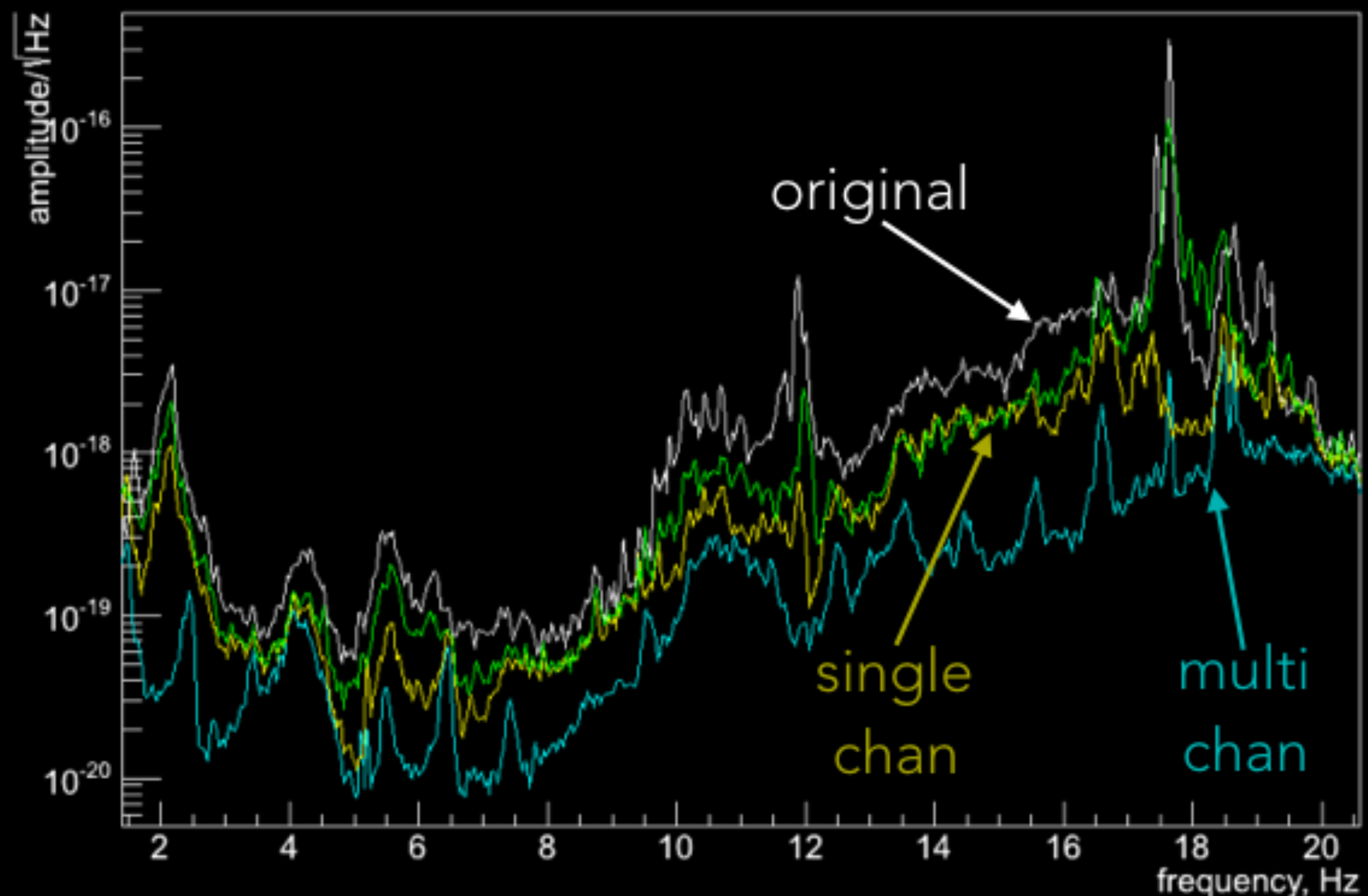


FIG. 2: Prediction of seismic noise using 16 coil current channel. Black curve is the original power spectrum. Red curve is obtained by conditioning the data using multichannel analysis. Blue curve is obtained by performing single channel regression analysis, on the target data, using the CC channels consecutively. Purple curve is a single channel analysis with a different order of the CC channels.

REGRESSION OF ENVIRONMENTAL NOISE IN LIGO DATA (LIGO-P1400004-V4)

Removal of bilinear noise from the up-conversion of low frequency noise with: power / calibration lines, violin modes

[...] bilinear coupling of the seismic noise which appears as side-bands around the power lines and calibration lines.

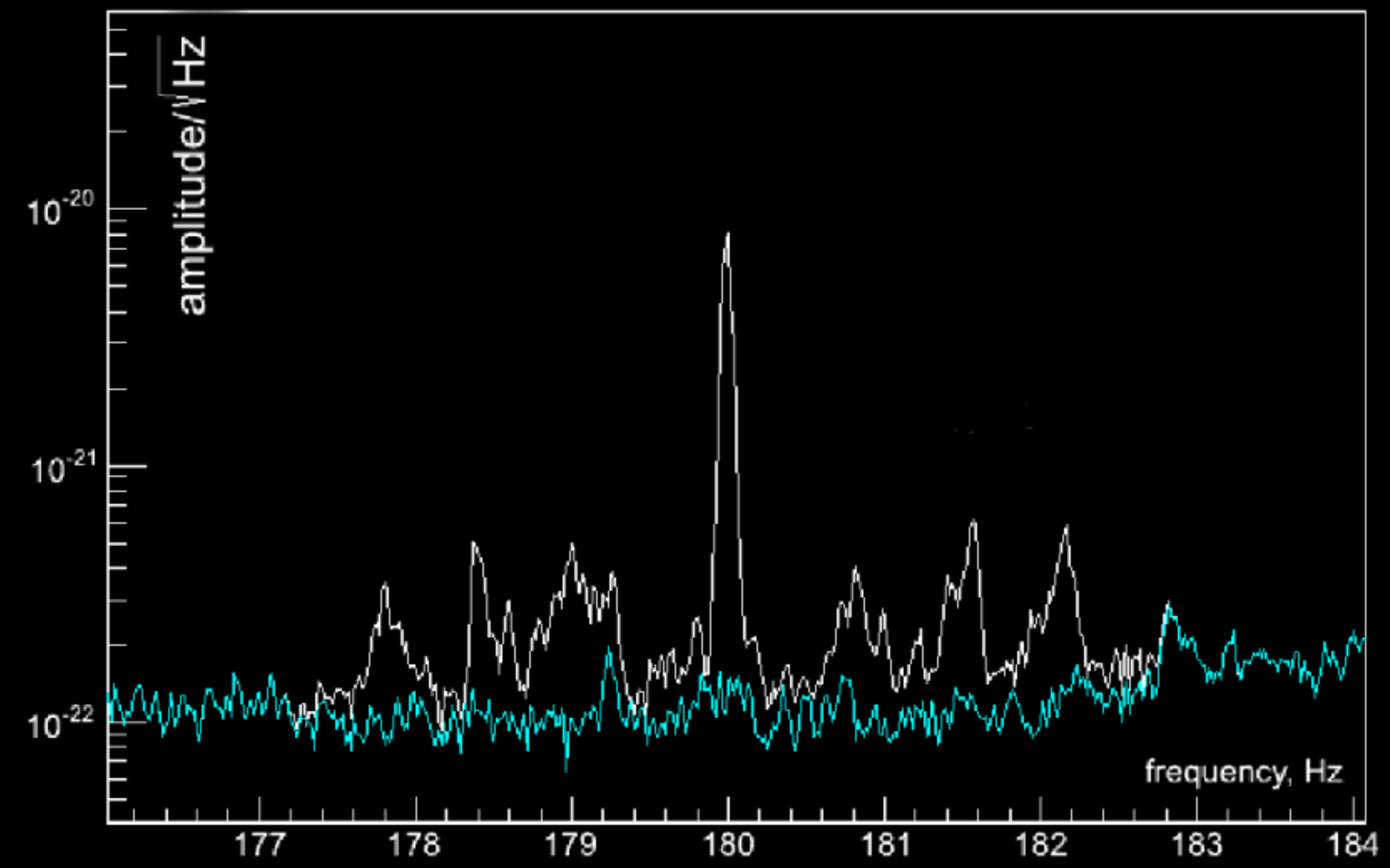


FIG. 8: Removal of the up-conversion noise around 180 Hz power line with 16 synthetic witness channels constructed from 16 coil current channels and the voltage monitor 6. Black curve is the original power spectrum. Red curve is obtained after conditioning the data.

VOLTERRA FILTER

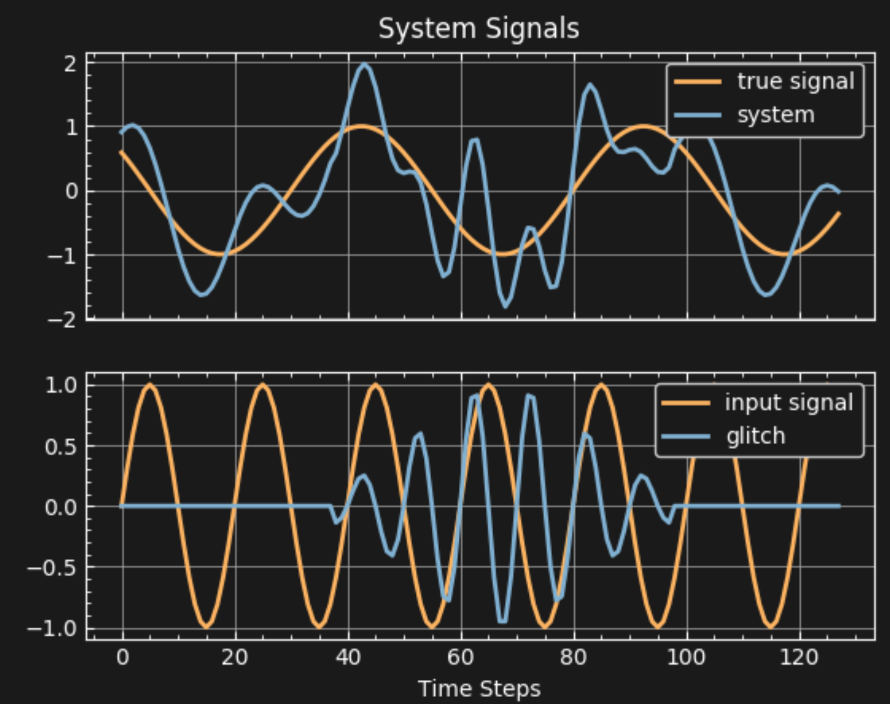
$$y[k] = a_0 + \sum_{n_1=0}^{N-1} a_1[n_1]w[k-n_1] + \sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} a_2[n_1,n_2]w[k-n_1]w[k-n_2]$$

- 2nd order N-tap non-linear adaptive (non-stationary) filter.
- Probably can assume that the data is wide-sense-stationary on longer time scales, so the filter coefficients are stationary
- On shorter time scales, we would do better to let the filter coefficients be adaptive
- MLAs don't seem to be able to figure this out very well

AF (NONSTATIONARY & LINEAR)

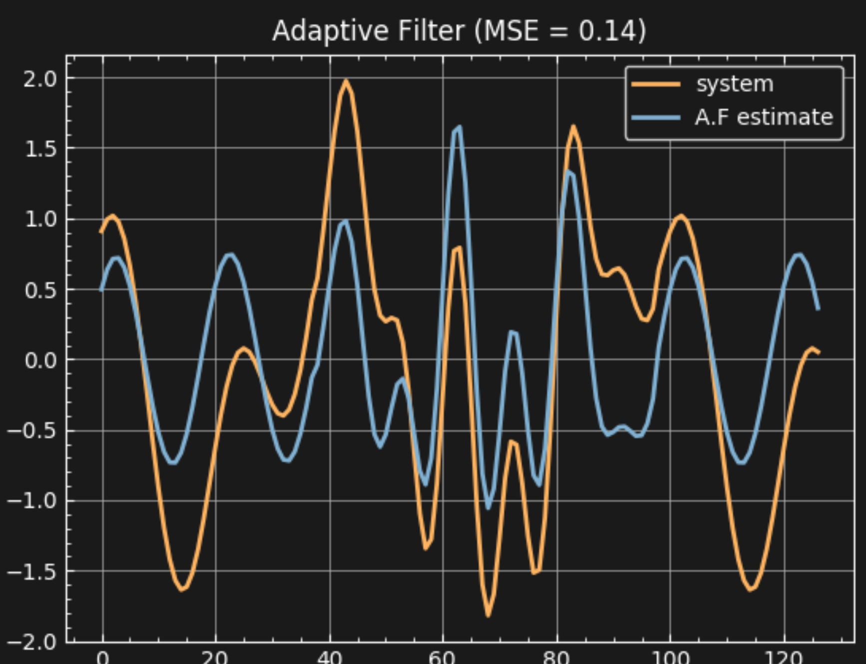
System

$$d[n] = x[n] + 0.2x[n-1]$$
$$-0.65x[n-2] + \text{wit}[n]$$
$$+g[n]$$

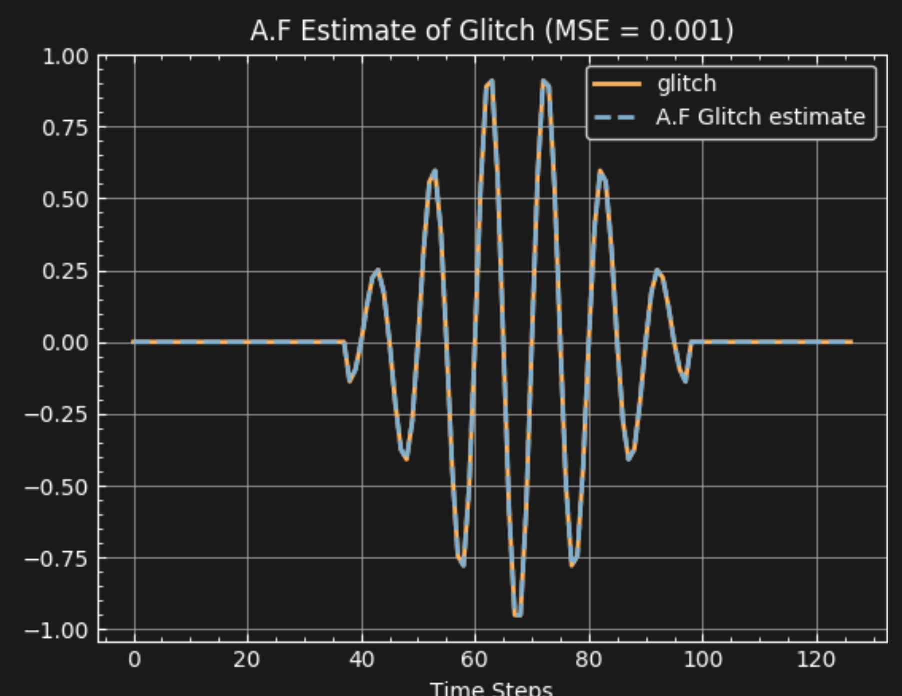


Input

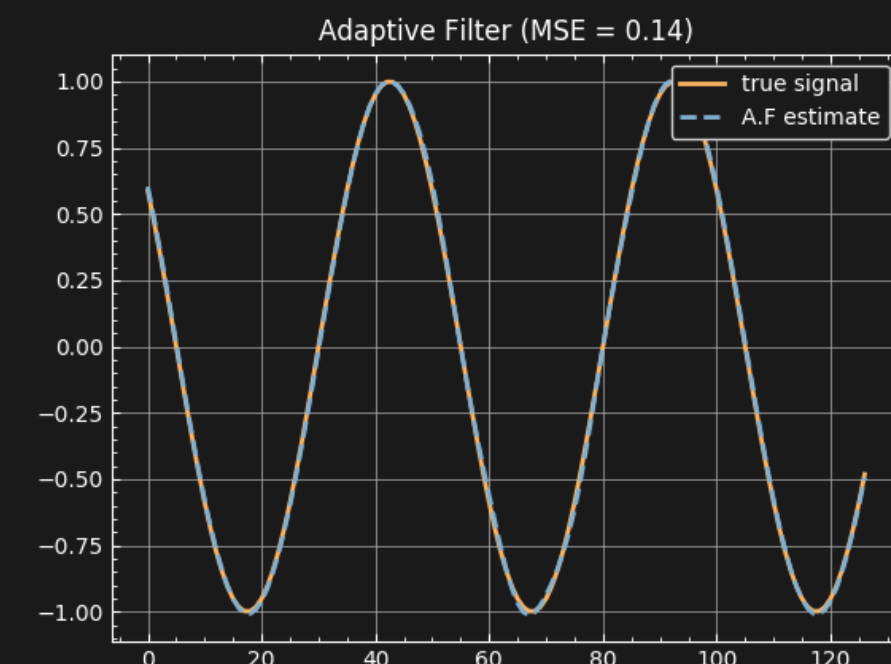
$$x[n], g[n]$$



$$\hat{y}[n] = f(x[n], x[n-1], x[n-2])$$



$$\hat{y}[n] = f(x[n], x[n-1], x[n-2])$$

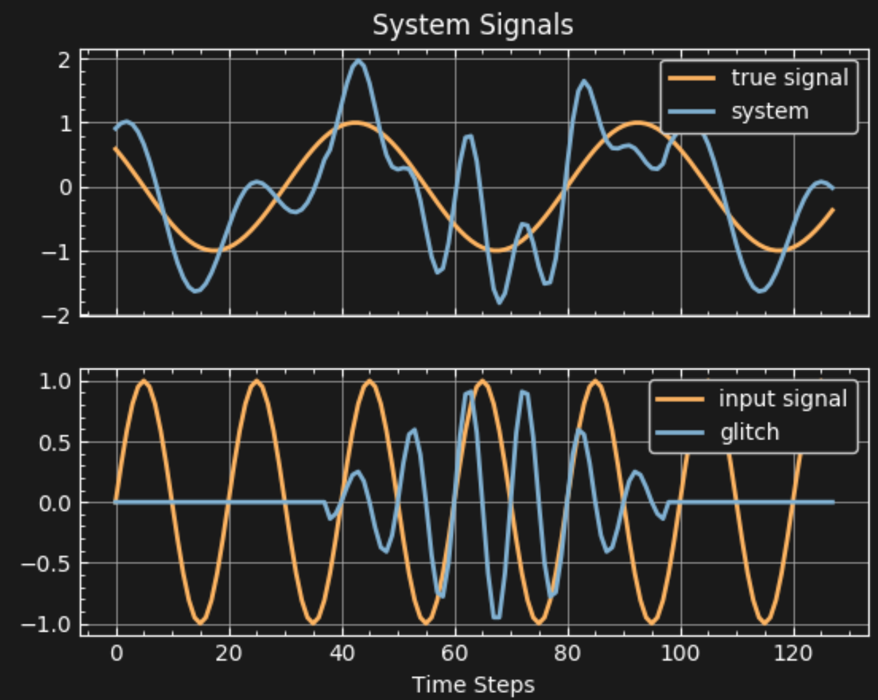


$$\hat{y}[n] = f(x[n], x[n-1], x[n-2])$$

MLA (NONSTATIONARY & LINEAR)

System

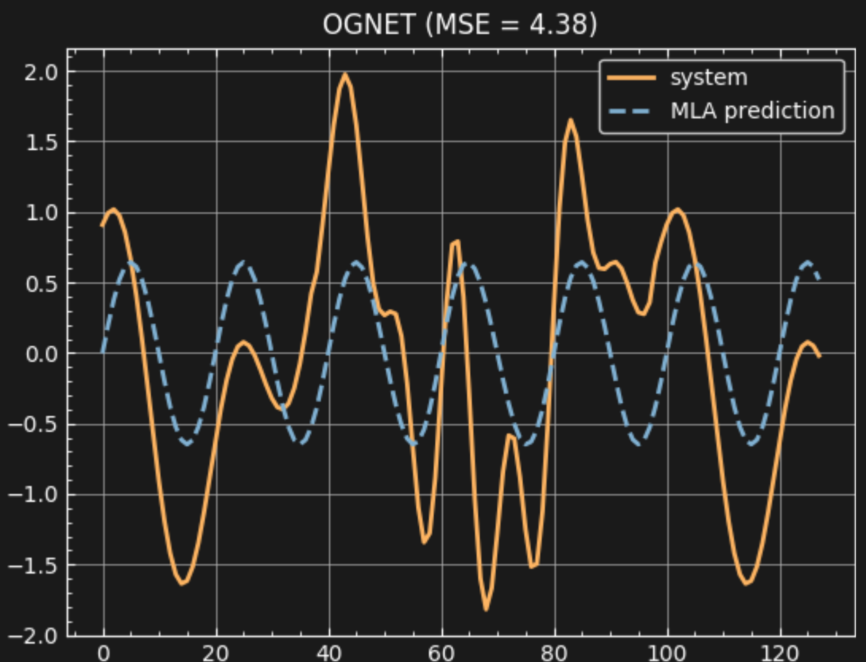
$$d[n] = x[n] + 0.2x[n-1]$$
$$-0.65x[n-2] + \mathrm{wit}[n]$$
$$+g[n]$$



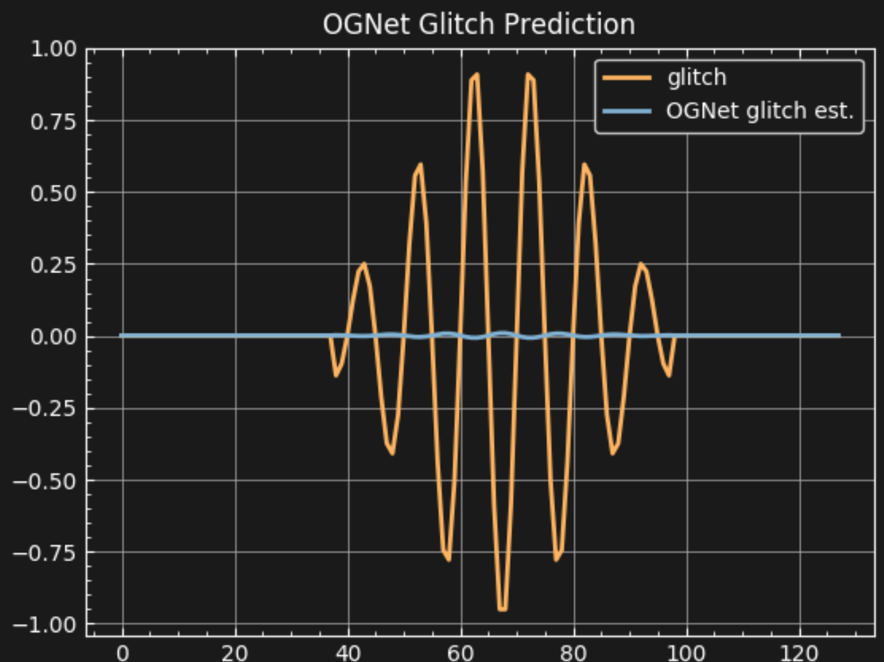
Input

$x[n], g[n]$

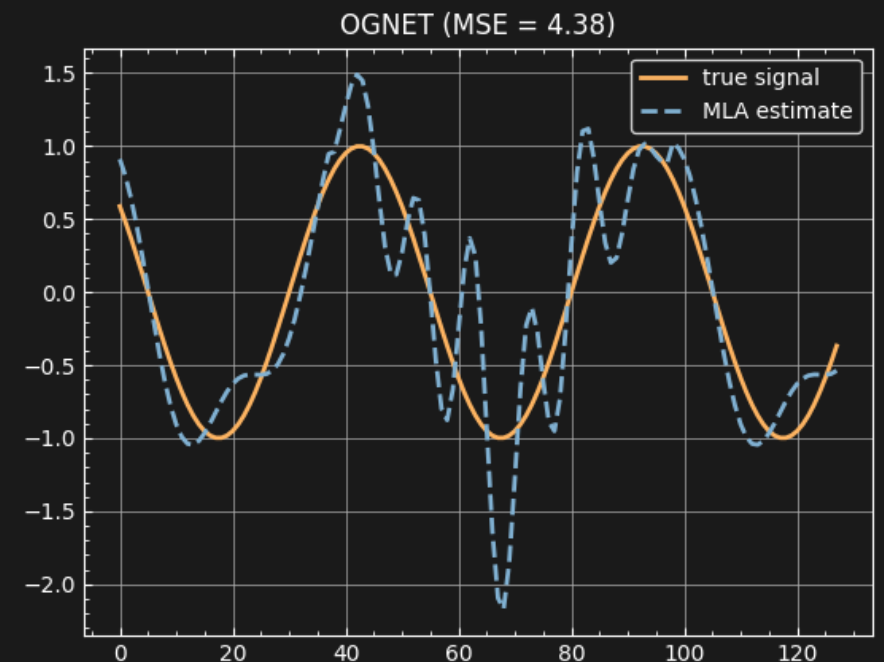
NS subtraction at the mercy of initialized values



$\hat{y}[n] = f(x[n], g[n])$



$\hat{y}[n] = f(g[n])$

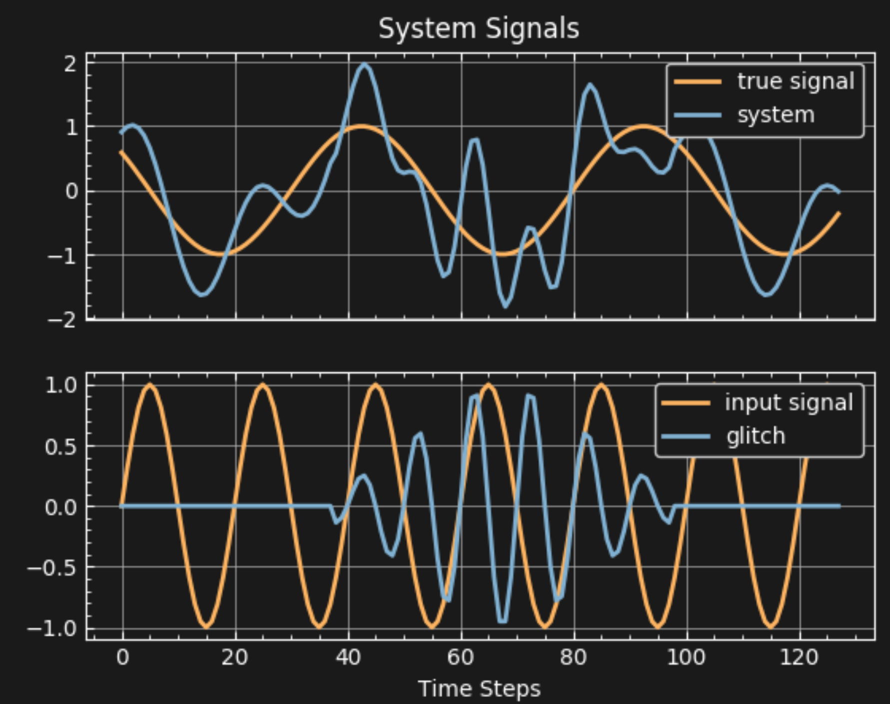


$\hat{y}[n] = f(x[n], g[n])$

AMLA (NONSTATIONARY & LINEAR)

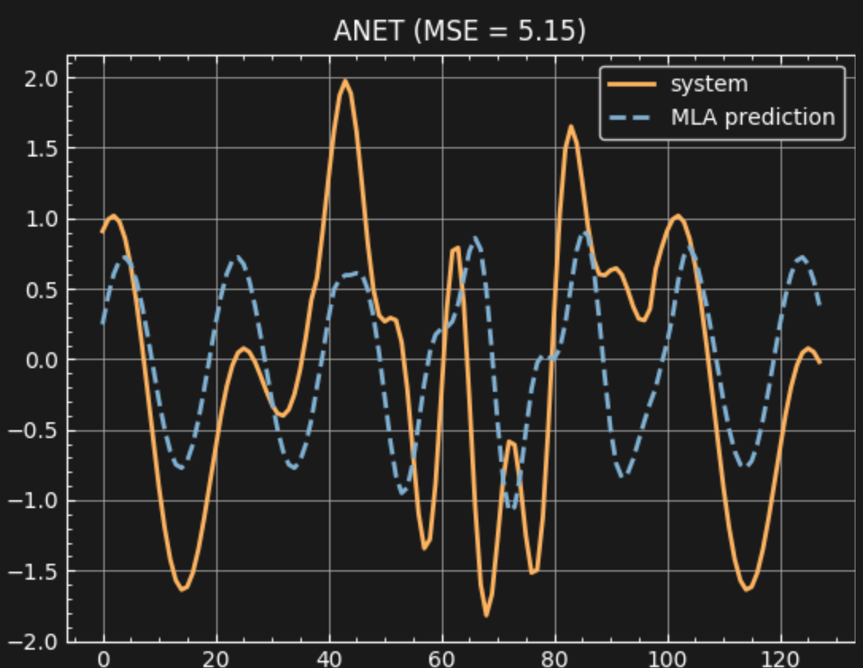
System

$$d[n] = x[n] + 0.2x[n-1]$$
$$-0.65x[n-2] + \mathrm{wit}[n]$$
$$+g[n]$$

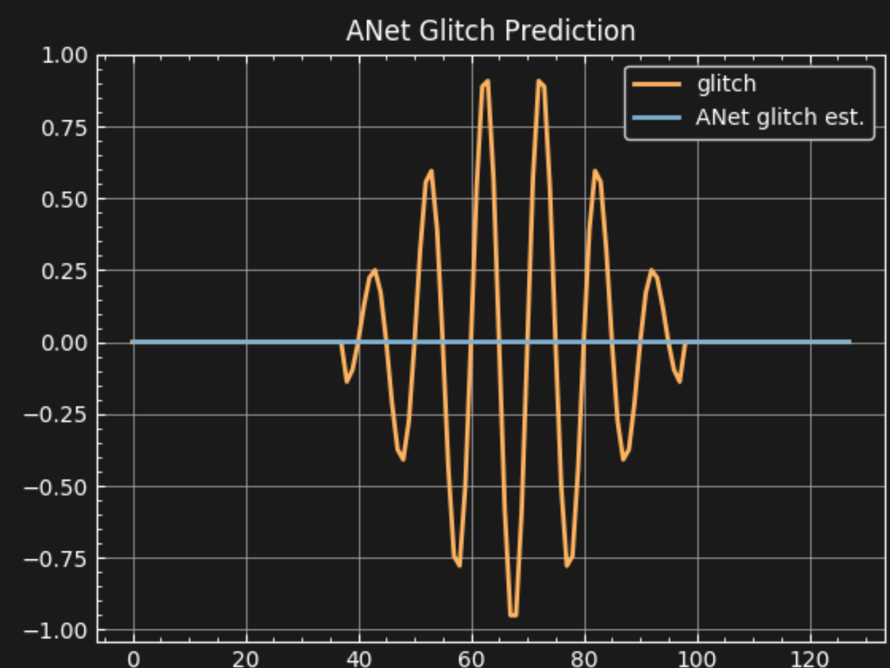


Input

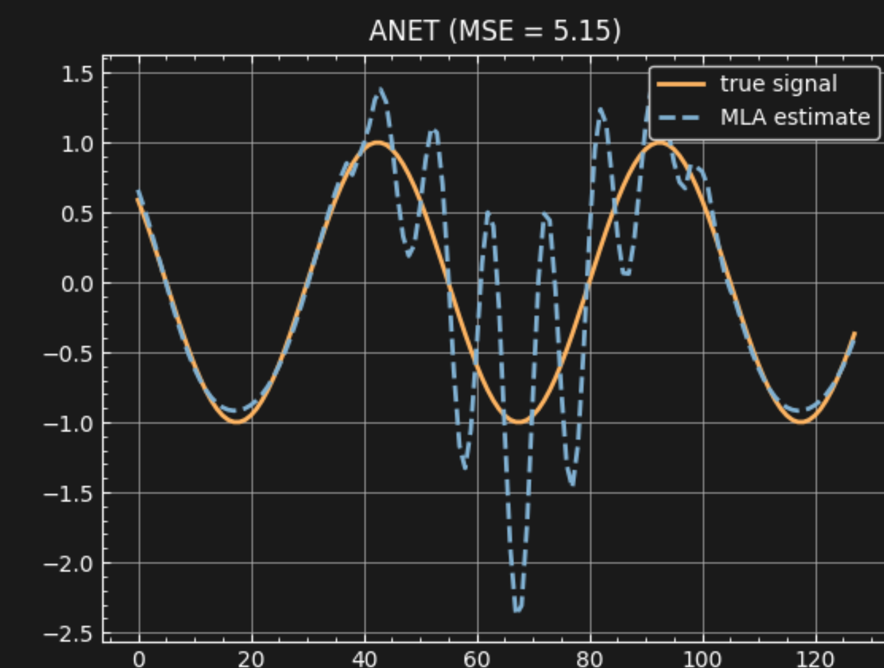
$$x[n], g[n]$$



$$\hat{y}[n] = f(x[n], \ldots x[n-1], g[n], \ldots, g[n-1])$$



$$\hat{y}[n] = f(g[n], g[n-1], g[n-2])$$

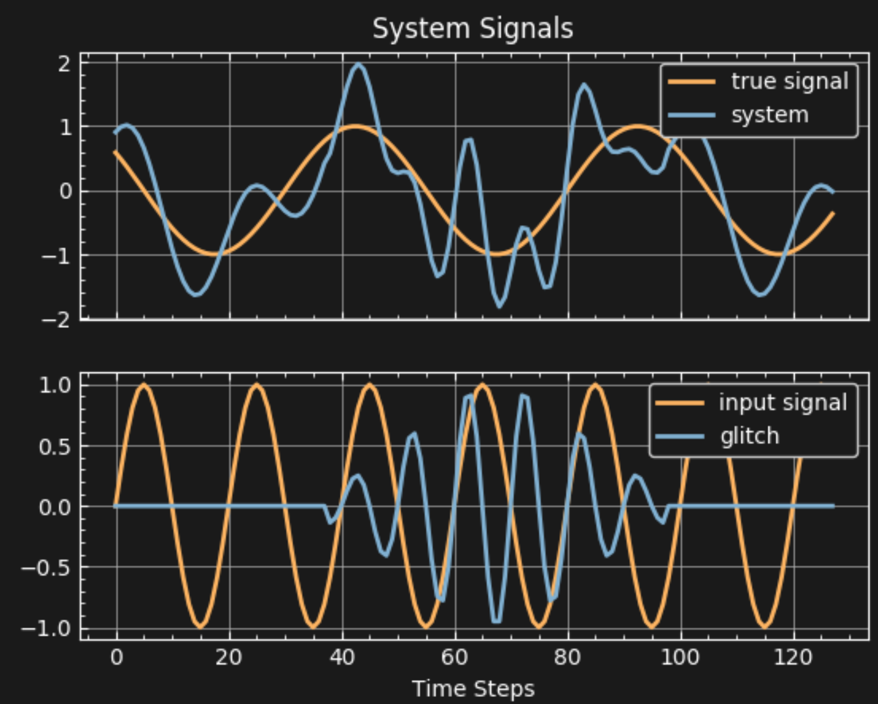


$$\hat{y}[n] = f(x[n], \ldots x[n-1], g[n], \ldots, g[n-1])$$

WHAT ABOUT LSTM NETWORKS?

System

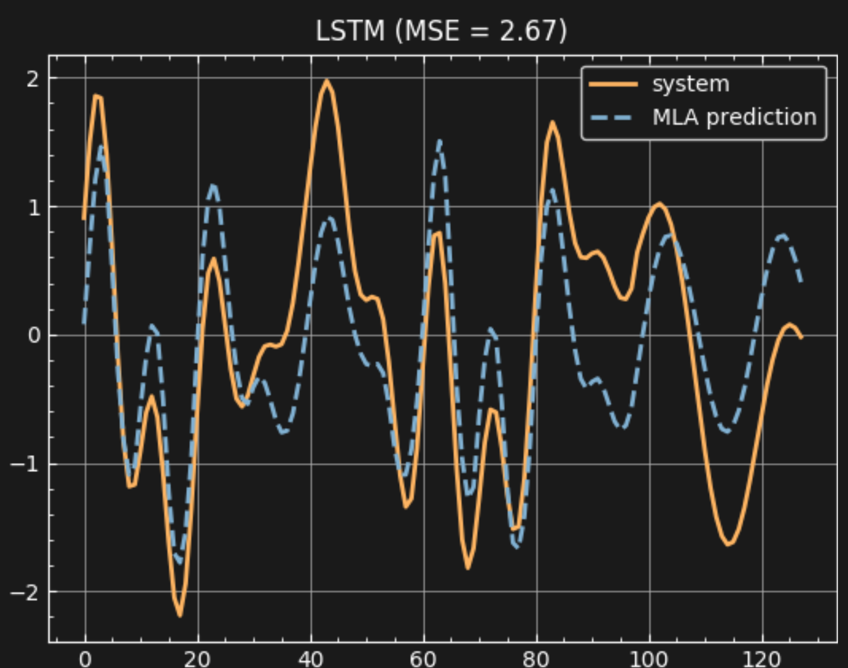
$$d[n] = x[n] + 0.2x[n-1]$$
$$-0.65x[n-2] + \mathrm{wit}[n]$$
$$+g[n]$$



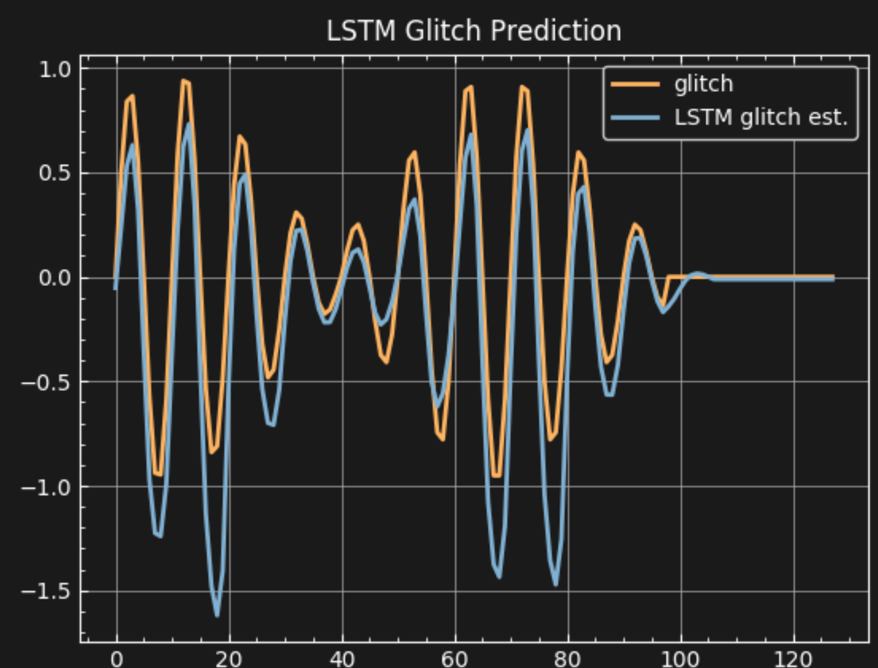
Input

$x[n], g[n]$

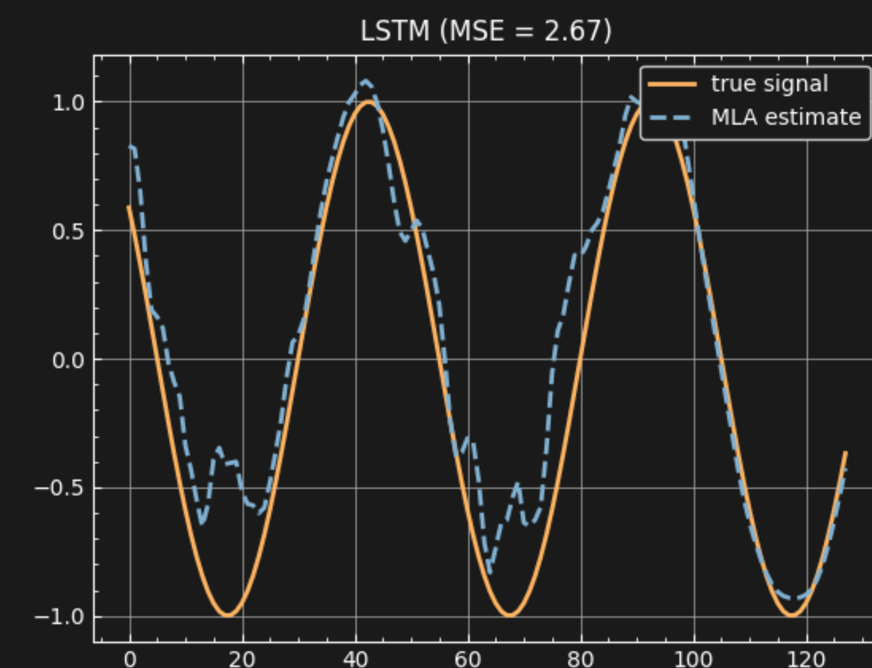
Only works if we can train on the nonstationarity



$\hat{y}[n] = f(x[n], \ldots x[n-1], g[n], \ldots, g[n-1])$

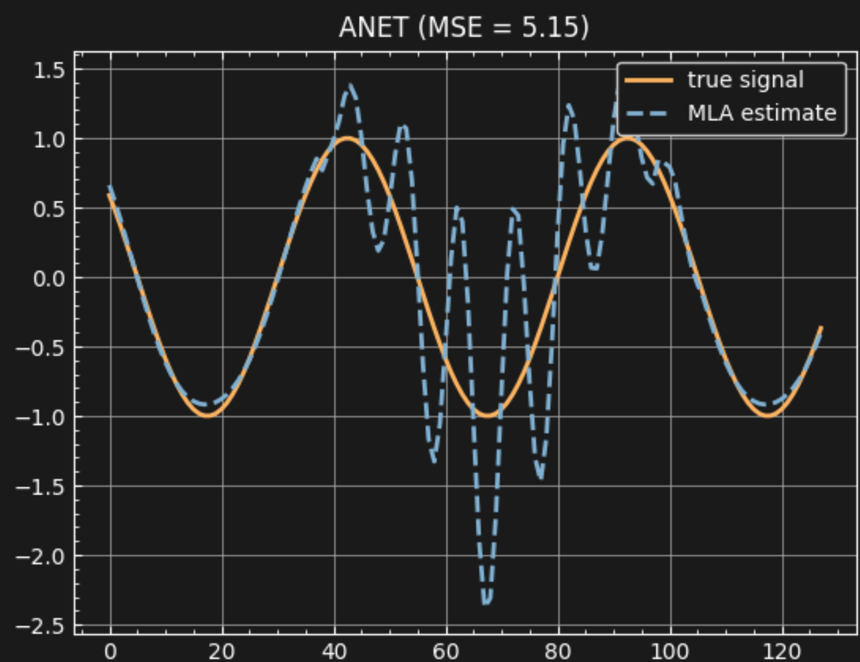


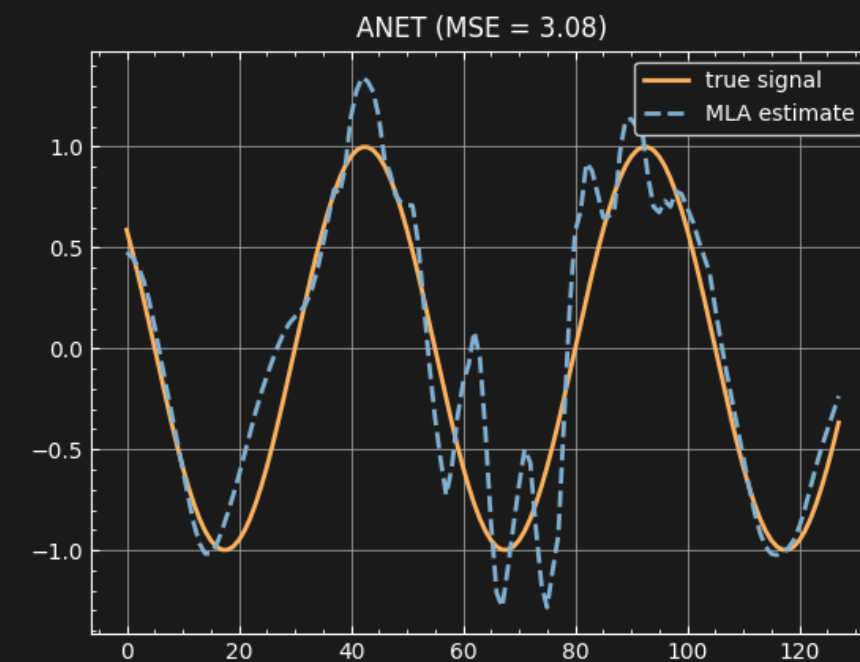
$\hat{y}[n] = f(g[n], g[n-1], g[n-2])$

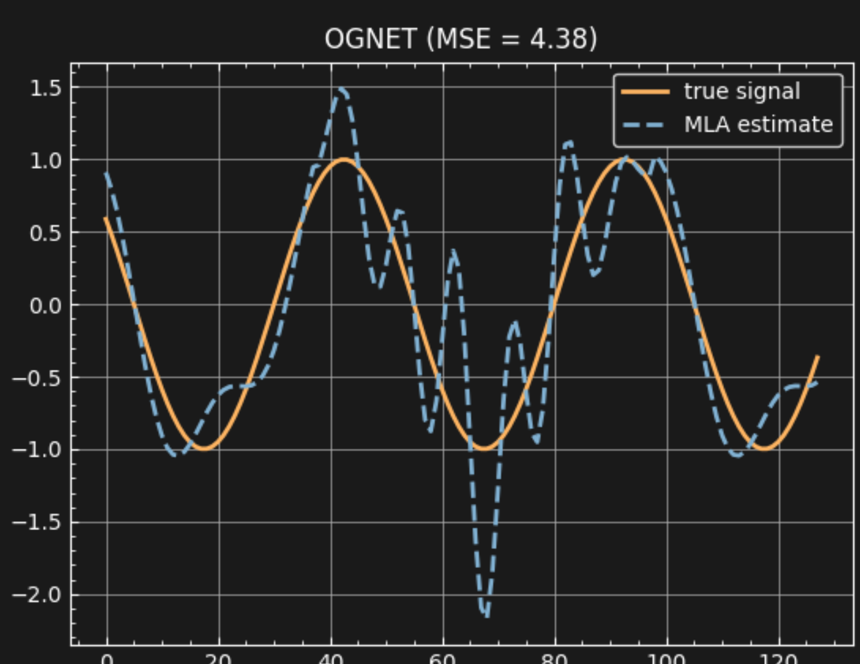


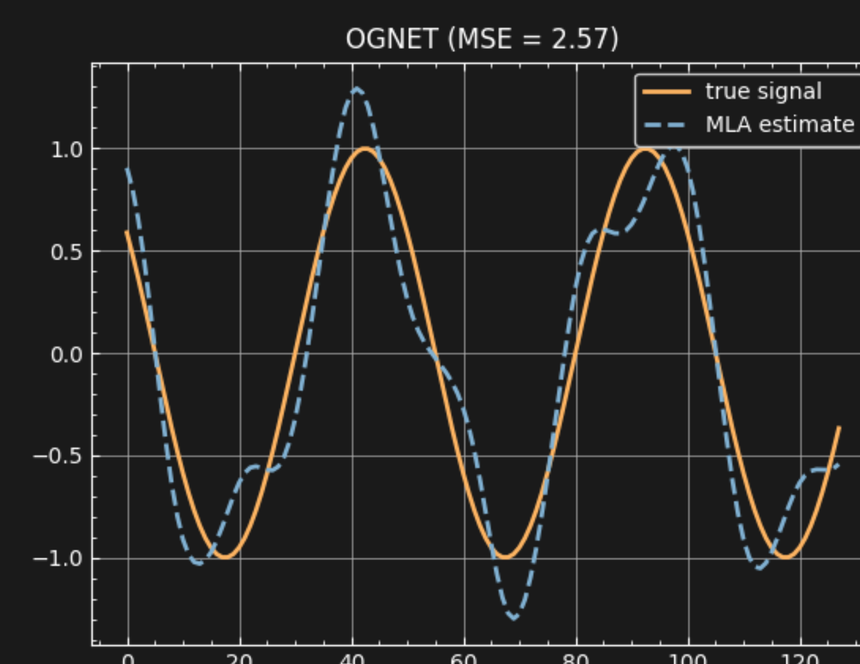
$\hat{y}[n] = f(x[n], \ldots x[n-1], g[n], \ldots, g[n-1])$

BUT THEN EVERYONE DOES BETTER

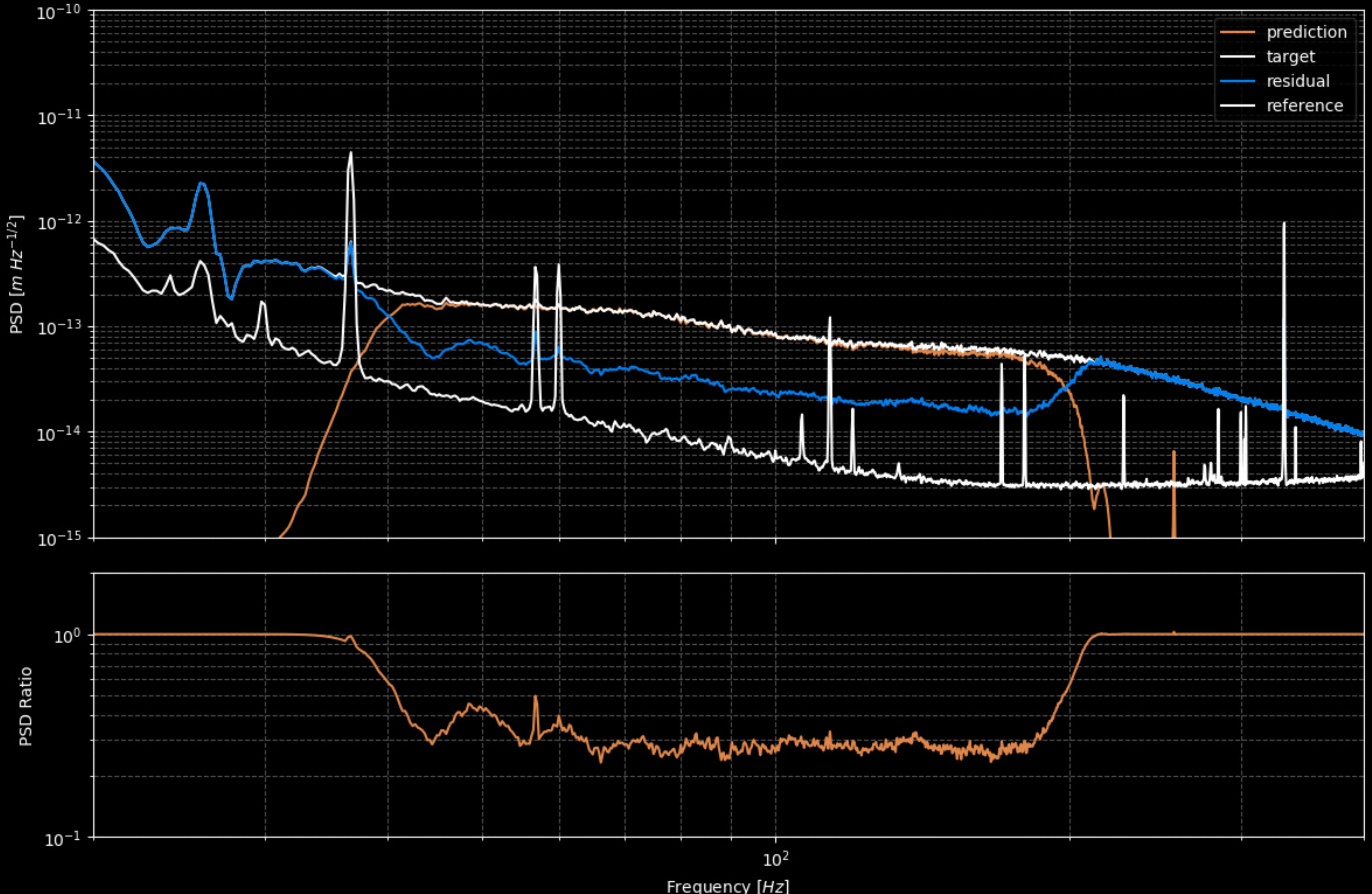








THIS COULD BE WHAT WE ARE MISSING



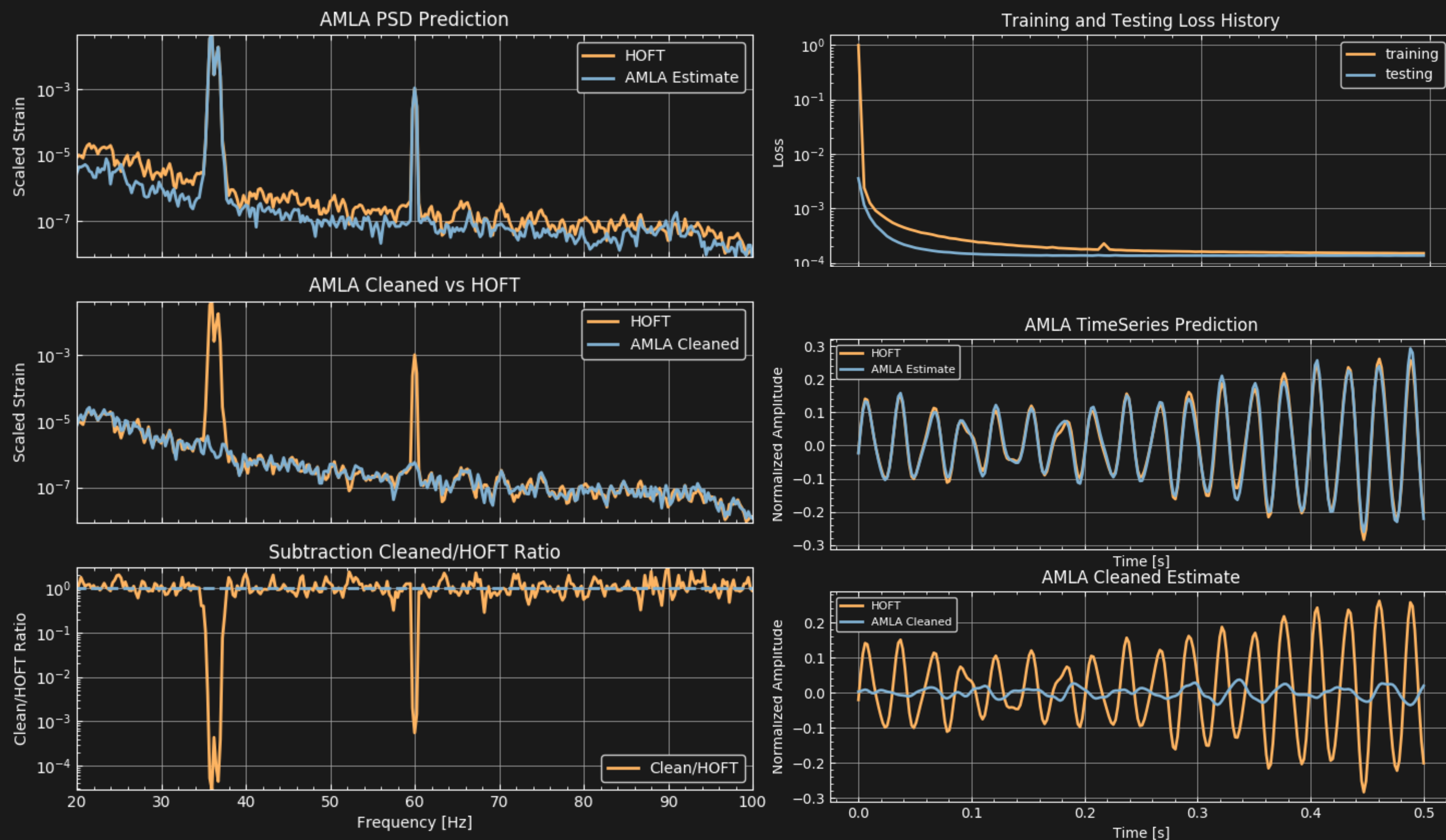
THIS COULD BE WHAT WE ARE MISSING

Convergence depends on the spread of eigenvalues of the autocorrelation matrices

$$\text{convergence} \approx \left(\frac{\frac{\lambda_{max}}{\lambda_{min}} - 1}{\frac{\lambda_{max}}{\lambda_{min}} + 1} \right)^k$$

for k iterations

ADAPTIVE MLA - NO BANDPASSING



SOME THINGS TO WORK ON

- Scale everything by its PSD (or better scaled loss function)
- Discrete cosine transform (pick out important spectral features - similar to how CNNs work)
- Subband adaptive filters (sort of tried this without knowing it with NNs)
- 2nd order volterra filters
- Writing