~~Wavelet Analysis~~ & Line Removal

Presented by S.Klimenko

University of Florida

- Outline
  - ~~Wavelets~~
    - ~~Wavelet Analysis Tool~~
    - ~~Current status~~
    - ~~Plans & Conclusion~~
  - Line removal
    - Algorithm
    - Results
    - Conclusion

LIGO-G000093-00-D

Coherent Lines Removal

- Wavelets algorithms can be used for time-frequency analysis of transients and/or short bursts of GW.
- Strong line interference produces a significant non-Gaussian noise masking other non-Gaussian components.
- Effective, simple and fast line removal algorithm is necessary for wavelet analysis.

Line Removal Algorithm

- DFT of data of N samples:
  - basis of orthogonal Fourier functions:

    $F_k(n)=e^{-2\pi i\, n\, k/N}$, $k,n = 0,..,N-1$; $\delta_{ij}=\Sigma_n F_i(n) F_j(n)$
  - sampled harmonic signal: $L(n)=a\, e^{-2\pi i\, n\, f/f_o}$,
    - f - harmonic signal frequency
    - f_o - sampling rate
  - if $f/f_o=k/N$, $L(n) = a_k F_k(n)$ - one of the basis Fourier functions.
- Removing of line and its harmonics $\{L_k(n)\}$.
  - resample data with sampling rate f_s: $f_s/f = int(f_o/f)+1$
  - select data sample length: $N = k\, f_s/f$, $k=1,2\ldots$
  - extract interference signal: $I(n) = \Sigma_k L_k(n)= \Sigma_k a_k F_k(n)$
  - re-sample $I(n)$ back & subtract from original data

Data re-sampling

- Sample rate converting
  - reconstruction: $s(n\Delta) \rightarrow s(t)$; $\Delta=1/f_0$
    ST: s(t) perfectly represents frequencies that are less then $f_0/2$
  - sample data at new sampling rate: $s(t) \rightarrow s'(n\Delta')$; $\Delta'=1/f_s$
- Data interpolation filter
  - wavelets
  - other interpolating techniques
  - currently n^{th} order polynomial interpolation filter is used (lifting wavelets use the same filter)
- Up-sampled ($f_s > f_0$) data used to find interference due to harmonic lines

Line Interference Signal

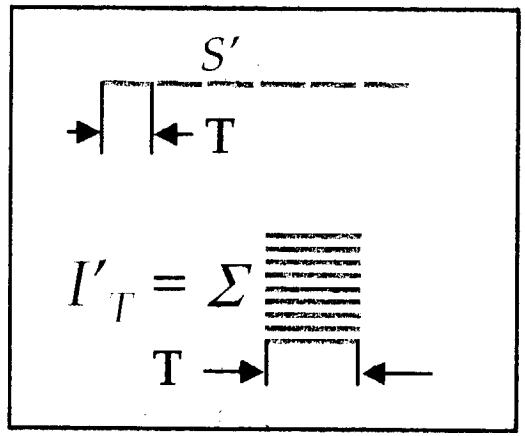
- line extraction

$$I'(n) = \Sigma_k \, a_k F_k^*(n) \, \phi_k$$

  - $a_k = \Sigma_n \, s'(n) \, F_k(n)$ - Fourier coefficient for k[th] harmonic
  - ϕ_k - optimal filter; $\phi_k = 1$, if neglect noise for k[th] harmonic

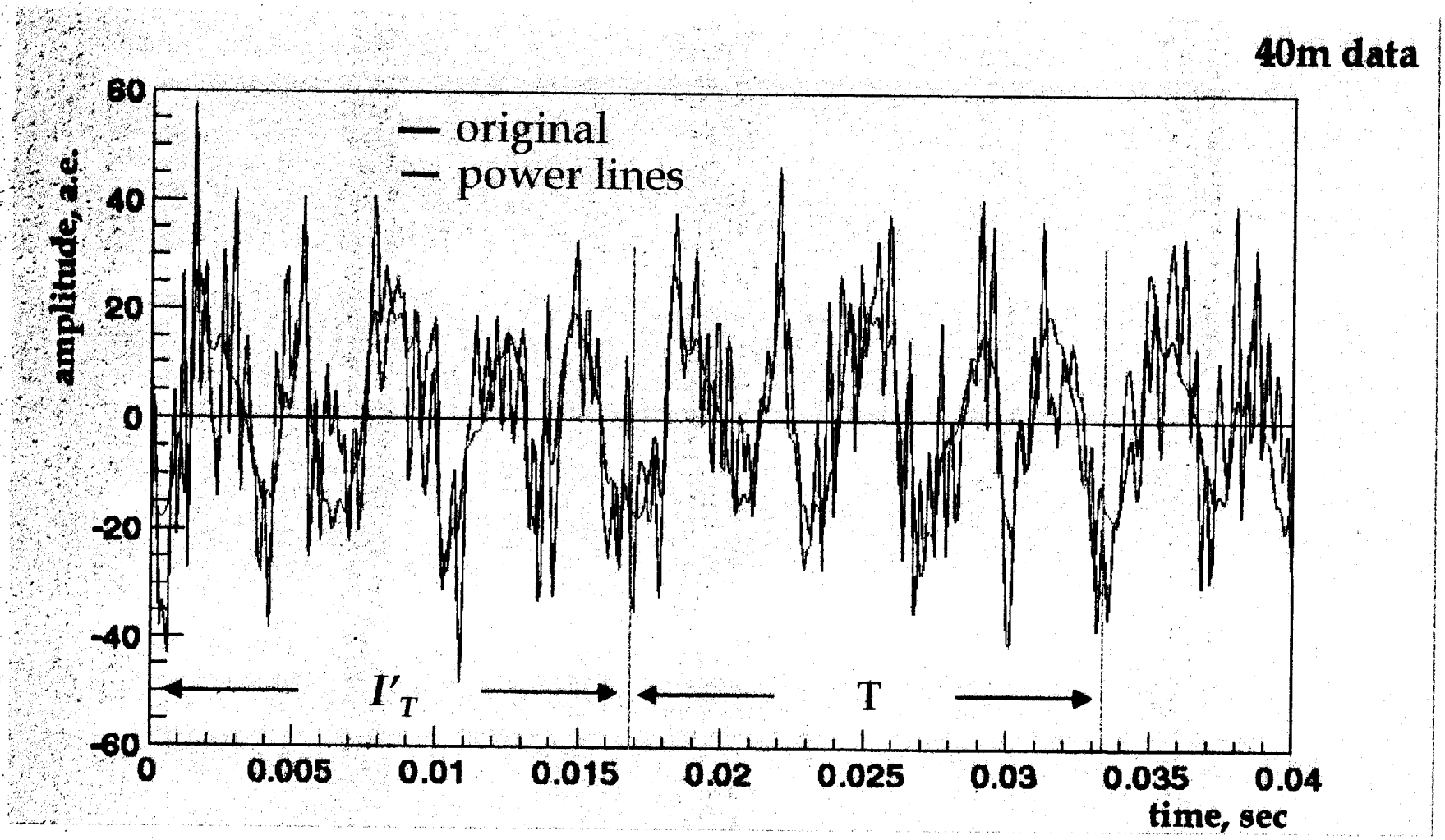
- fast line extraction in T domain ($\phi_k = 1$) :

  - T - period of fundamental harmonic
  - I'_T - one period of $I'(n)$ for all harmonics
  - save I'_T along with filtered signal to recover original signal ~160 for PL



  - signals $s'(n)$-$I'(n)$ and $I'(n)$ are orthogonal by definition
  - for optimal filtering a_k can be found by Fourier transform of I'_T

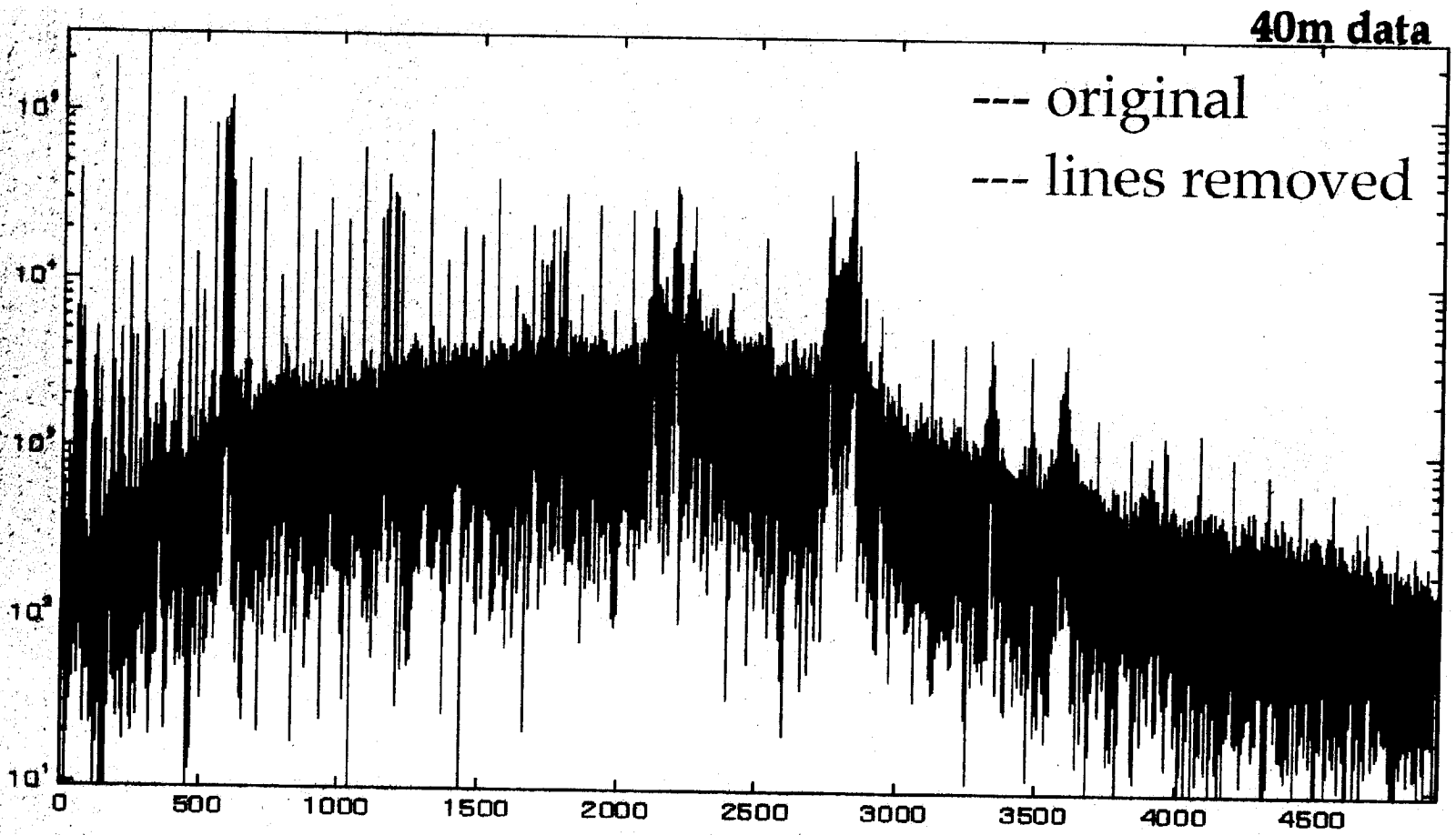
Power Lines Interference



S.Klimenko

Power Lines Removal

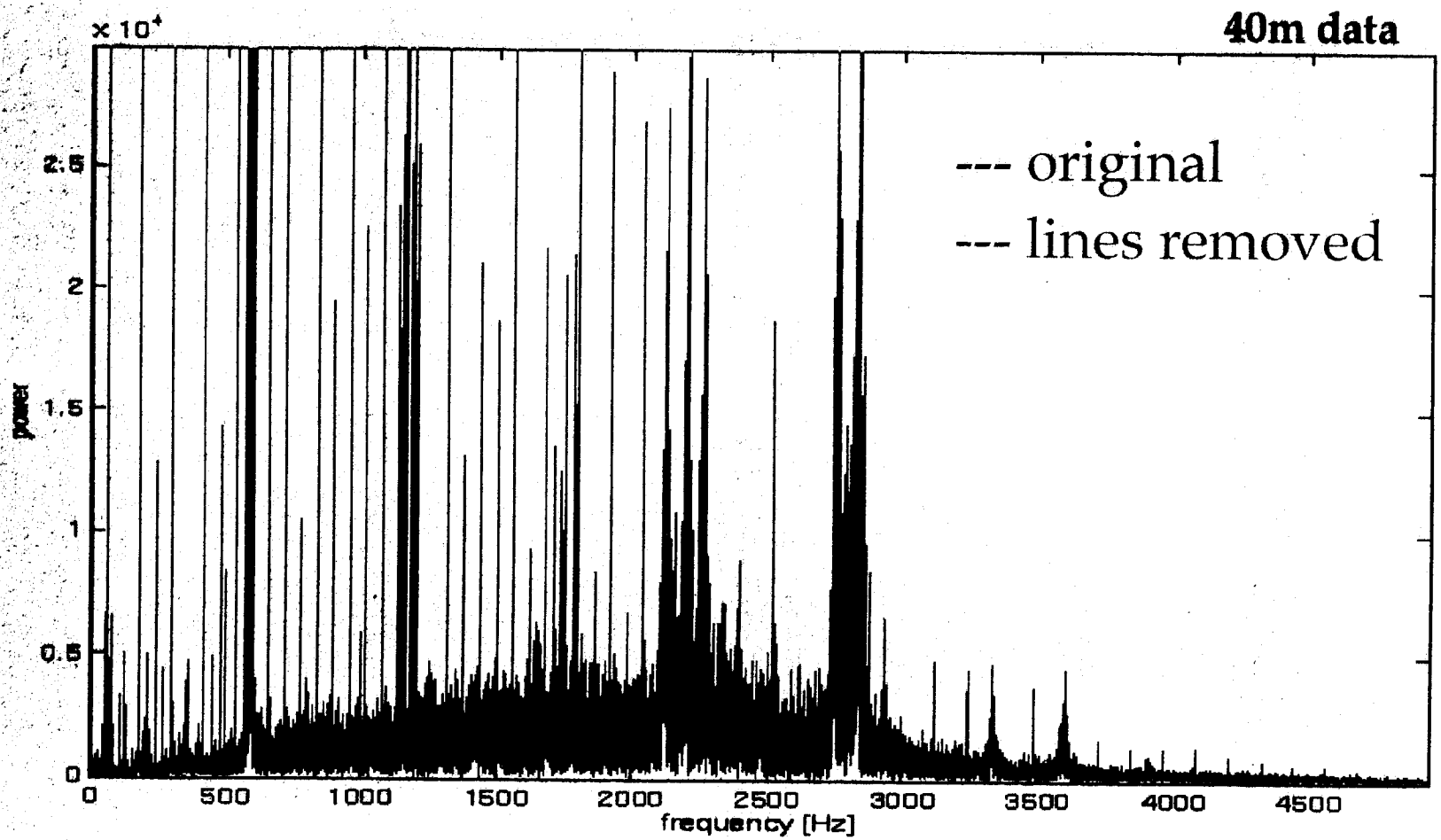
- Blue - Fourier spectra with power lines, red - Fourier spectra with power lines removed. (T=5sec)



S.Klimenko

Power Lines Removal

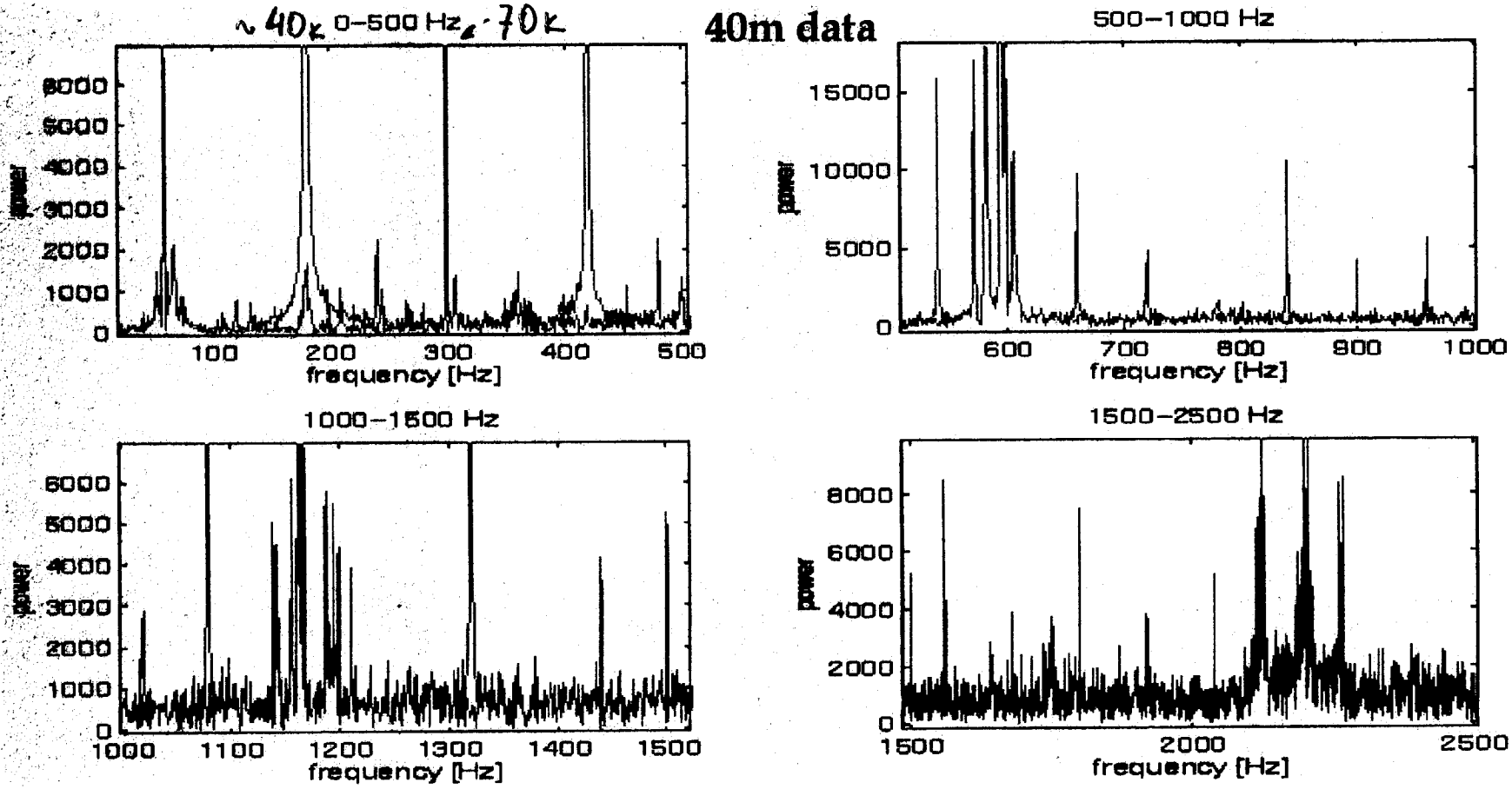
- blue - Fourier spectra with power lines, red - Fourier spectra with power lines removed. (T=5sec)



S.Klimenko

Power Lines Removal

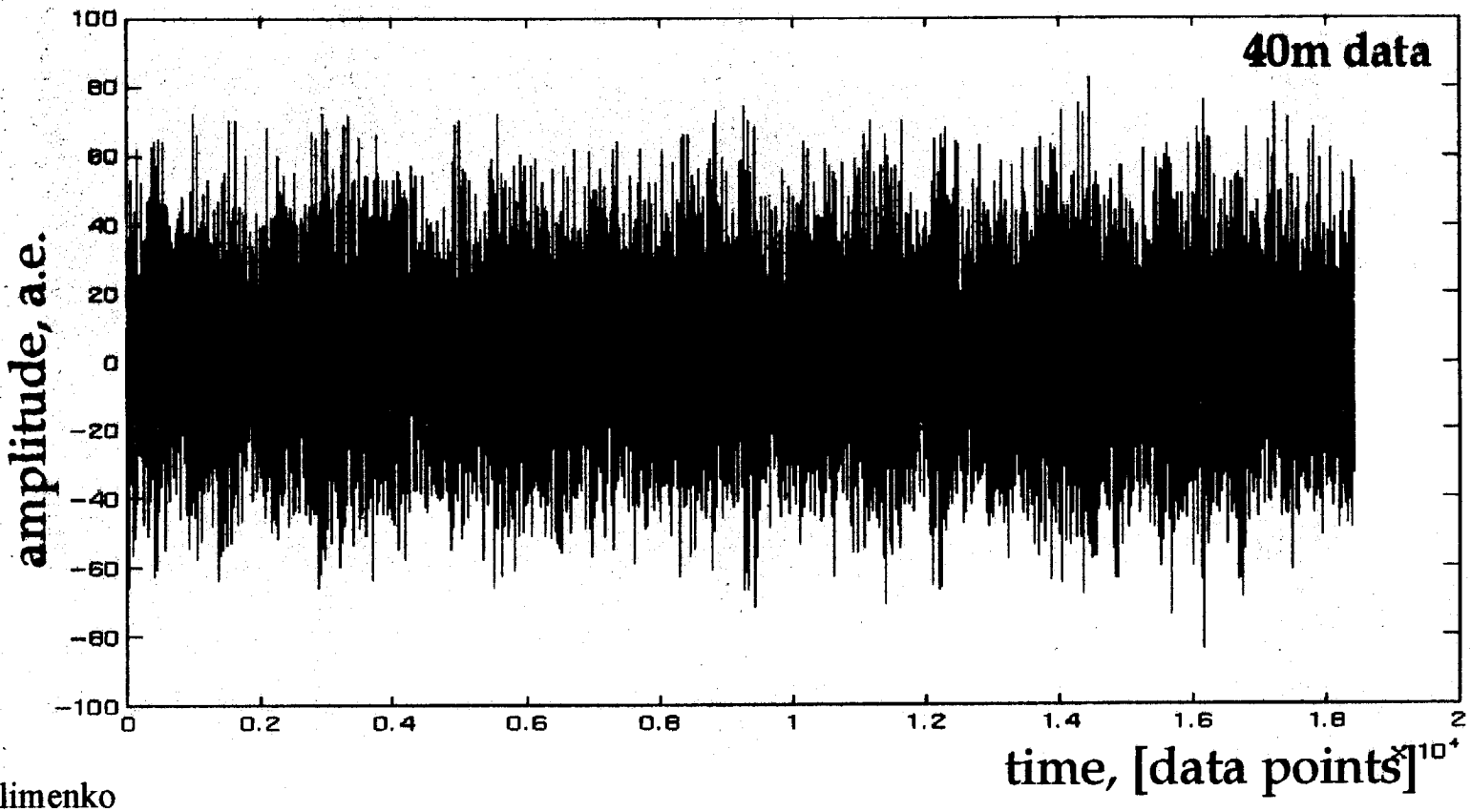
- Sample of 10000 points, (1sec), blue - data with lines, red - lines removed



S.Klimenko

Lines Removal

- blue - original data, red - 60Hz lines removed, green - 582.4Hz lines removed. Signal energy: $23.4^2 : 17.5^2 : 10.8^2$
- Energy balance: $\delta E = \langle s^2 \rangle - \langle I^2 \rangle - \langle (s-I)^2 \rangle$, $\delta E / \langle s^2 \rangle \sim 10^{-4}$



S.Klimenko

Power Lines Statistics

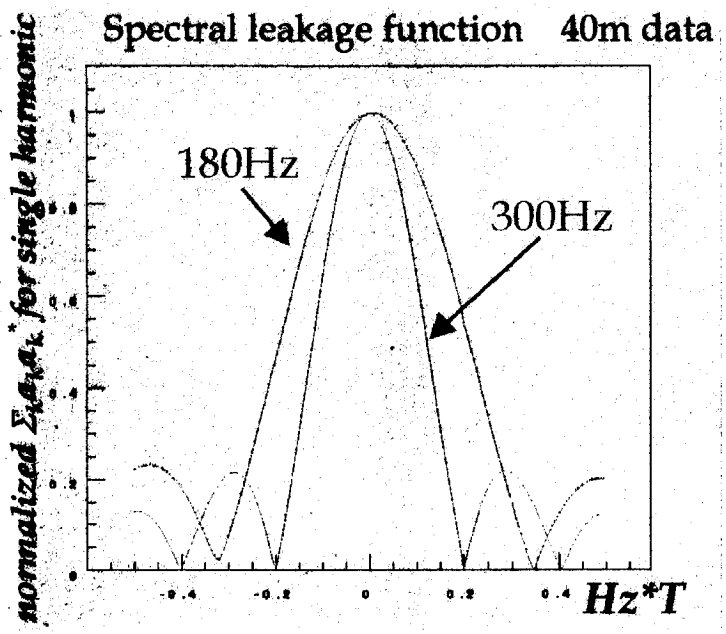
#	frequency	A+N	K
1	60.00190	0.191	0.24
2	120.0038	0.096	9.45
3	180.0057	4.074	40.71
4	240.0076	0.261	14.82
5	300.0095	7.223	107.37
6	360.0114	0.078	0.37
7	420.0133	2.344	23.78
8	480.0152	0.299	7.33
9	540.0171	1.663	58.76
10	600.0190	1.033	0.86
11	660.0209	0.989	13.61
12	720.0228	0.619	5.87
13	780.0247	0.159	2.22
14	840.0266	1.045	9.55
15	900.0285	0.472	12.66
16	960.0304	0.673	9.64
17	1020.032	0.472	5.79
18	1080.034	1.343	11.26
19	1140.036	0.591	4.33
20	1200.038	0.427	2.58
21	1260.039	0.111	1.36
22	1320.041	1.667	16.98
23	1380.043	0.322	1.80
24	1440.045	0.381	3.95
25	1500.047	0.445	4.08
26	1560.0494	0.776	16.16
27	1620.0513	0.203	1.15
28	1680.0532	0.453	2.87
29	1740.0551	0.164	1.00
30	1800.0570	0.706	15.49
31	1860.0589	0.204	2.20
32	1920.0608	0.684	6.87
33	1980.0627	0.116	1.07
34	2040.0646	0.667	5.39
35	2100.0665	0.210	0.85
36	2160.0684	0.264	1.54
37	2220.0703	0.230	0.69
38	2280.0722	0.123	0.50
39	2340.0741	0.069	0.60
40	2400.0760	0.253	2.69
41	2460.0779	0.087	0.53
42	2520.0798	0.366	1.92
43	2580.0817	0.037	0.45
44	2640.0836	0.049	0.52
45	2700.0855	0.081	0.60
46	2760.0874	0.163	0.29
47	2820.0893	0.469	0.49
48	2880.0912	0.215	3.06
49	2940.0931	0.034	0.17
50	3000.0950	0.123	1.45
51	3060.0969	0.0109	0.278
52	3120.0988	0.0858	2.285
53	3180.1007	0.0172	1.007
54	3240.1026	0.1175	5.884
55	3300.1045	0.0393	2.658
56	3360.1064	0.0138	0.276
57	3420.1083	0.0148	0.717
58	3480.1102	0.0836	2.740
59	3540.1121	0.0150	0.637
60	3600.1140	0.0702	0.549
61	3660.1159	0.0124	1.204
62	3720.1178	0.0299	2.684
63	3780.1197	0.0066	0.535
64	3840.1216	0.0244	1.617
65	3900.1235	0.0307	2.129
66	3960.1254	0.0297	2.271
67	4020.1273	0.0044	0.846
68	4080.1292	0.0252	3.508
69	4140.1311	0.0053	0.675
70	4200.1330	0.0225	1.975
71	4260.1349	0.0072	1.042
72	4320.1368	0.0056	0.690
73	4380.1387	0.0087	2.579
74	4440.1406	0.0078	1.653
75	4500.1425	0.0016	0.272

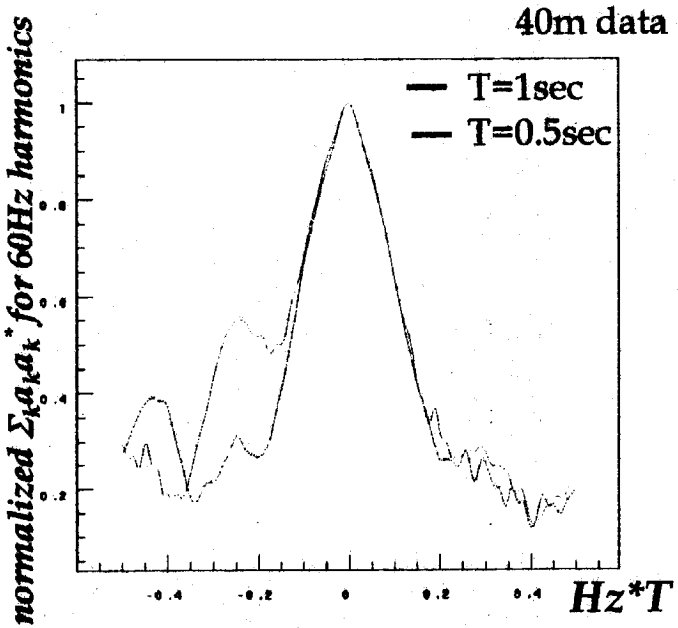
S.Klimenko

30%.

Line Frequency

- To use line removal algorithm, an accurate prediction of line fundamental frequency f is required.
- f estimation:
  - DFT of data gives rough estimate of f: $\delta f \sim f_0/N = 1/T$
  - re-sample data for given f and find harmonic amplitudes a_k
  - tune f to maximize $\Sigma_k a_k a_k^*$ for all (or group of) harmonics

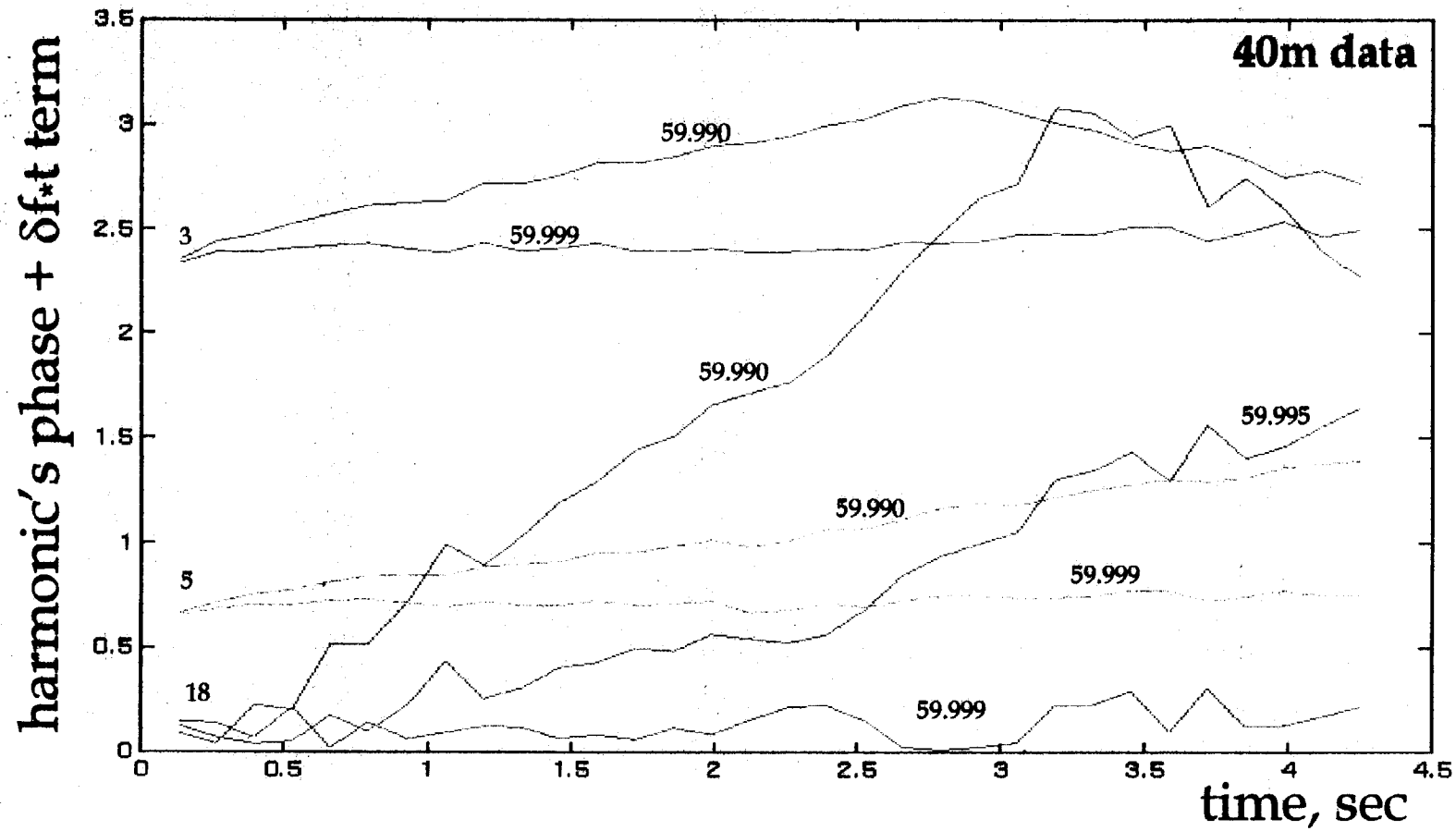




S.Klimenko

Power Line Fundamental Frequency

- 3,5,18 harmonic's phase ($\delta f_* t$ term) for 32 samples of data (0.132 sec each) for different fundamental frequencies.

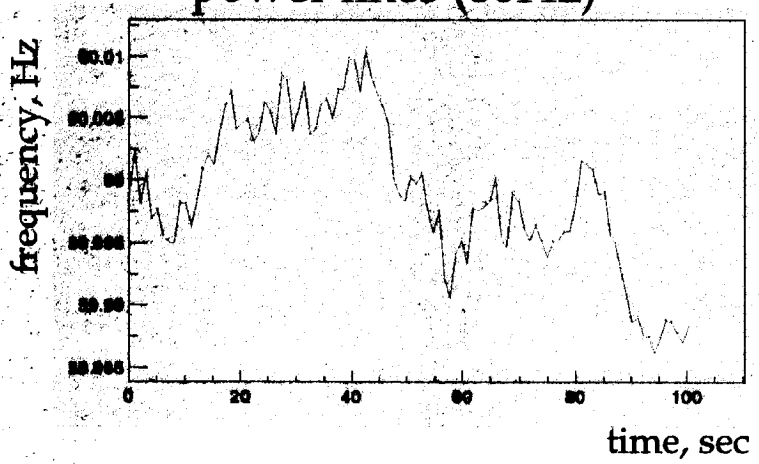


S.Klimenko

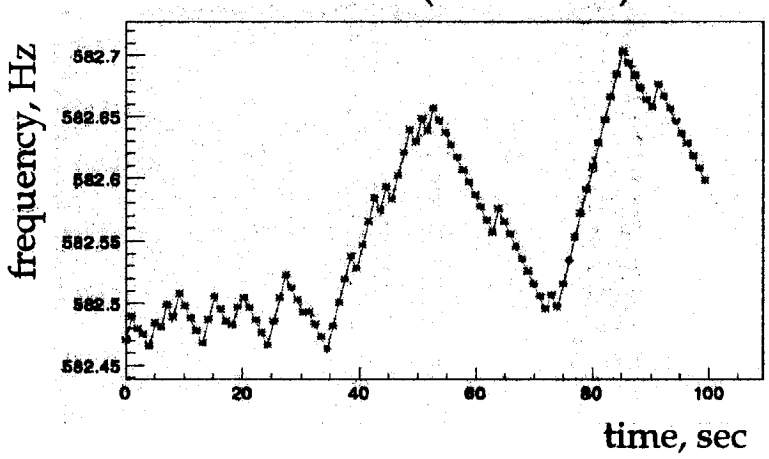
Monitoring of harmonic lines

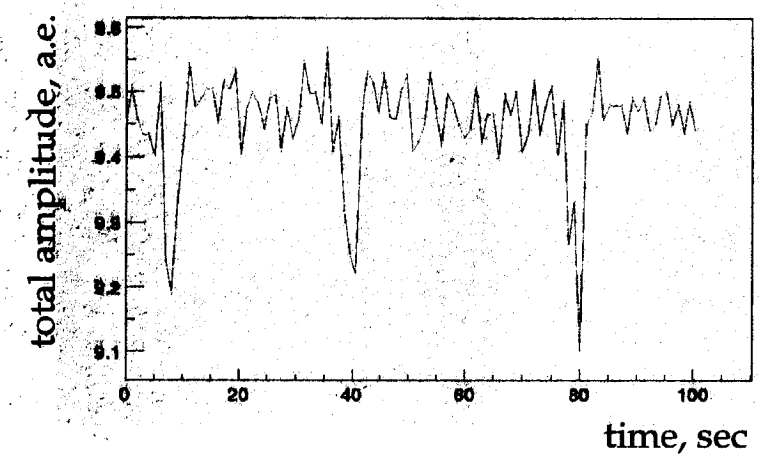
- 100 sec stretch of 40m data (20 frames)

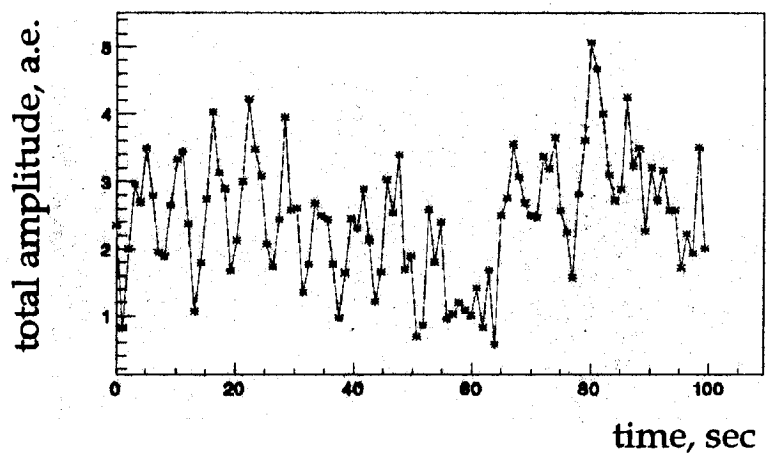
power lines (60Hz)



Violin lines (582.4 Hz)







S.Klimenko

Line Removal Software

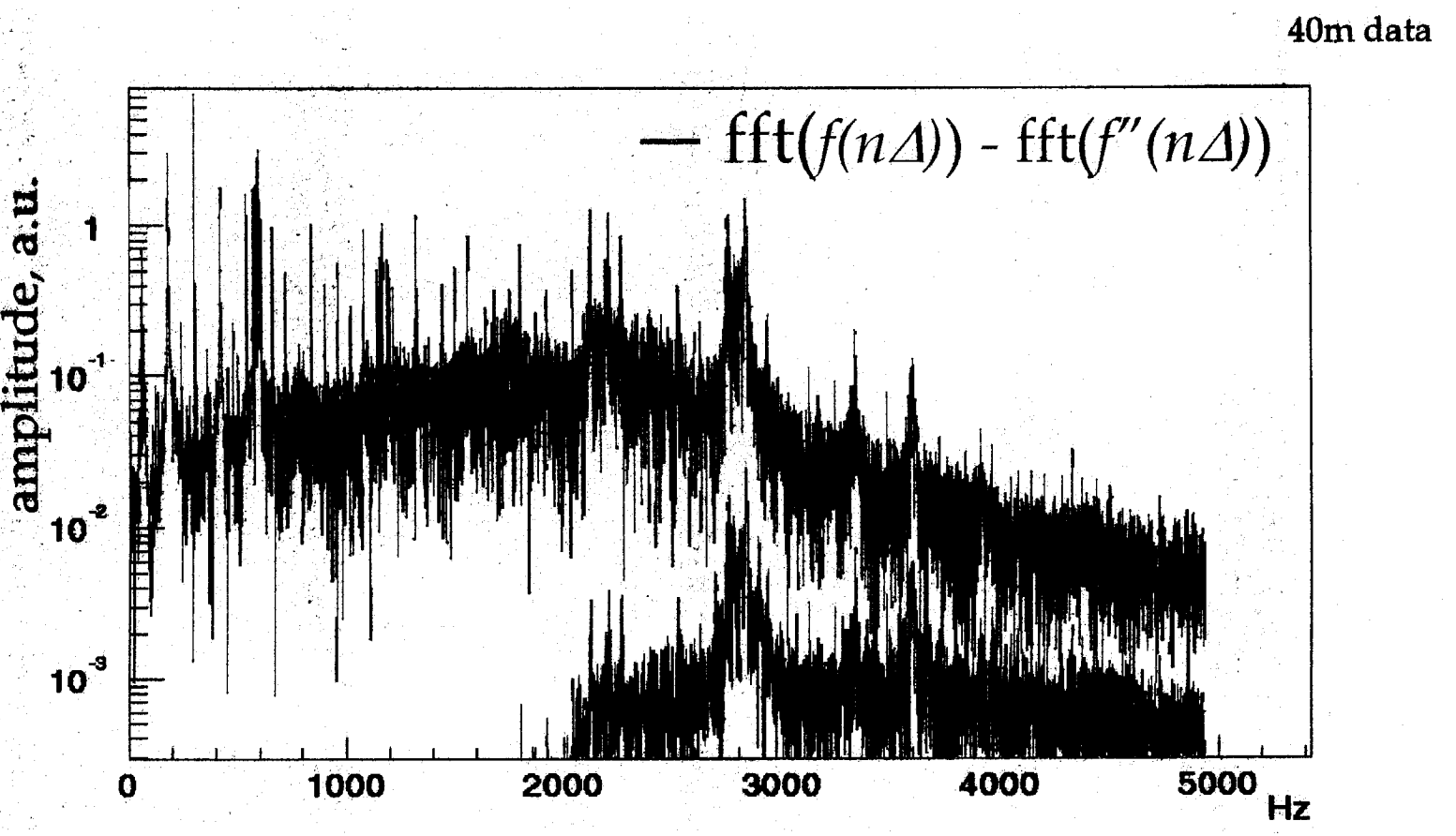
- **s.resample(s', f_s)**
  - re-sample data s' with sampling rate f_s .
  - uses polynomial interpolation
- **s.extract(f, n, m, E)**
  - extract n-m harmonics of frequency f from data s.
  - E - total energy of harmonics n-m
  - uses constant weight function
- **s.tune(f, n, m)**
  - tune fundamental frequency f maximizing $E=\Sigma_n a_k a_k^*$ for harmonics n-m and data set s
  - seed value of f can be taken from previous data set.
- LRS can be used for E, f monitoring & fast lines removal.
- LRS could be included into the Data Monitoring Tool (?)

Conclusion

- Fast and simple algorithm for harmonic lines removal is presented
- Single harmonic, group of harmonics or all harmonics of fundamental frequency f can be removed in one shot.
- Relatively small amount of information need to be saved to recover original data.
- Line removal software has been developed (C++ class-library). It can be used
  - to reduce non-Gaussian noise in the data
  - to monitor harmonic lines.

Re-sampling artifacts

$$f(n\Delta) \xrightarrow{-f_0} f'(n\Delta') \xrightarrow{-f_s} f''(n\Delta);$$



S.Klimenko

Note 1, Linda Turner, 05/09/00 10:13:01 AM
LIGO-G000093-00-D