Analysis of the LNS Implementation of the Fast Affine Projection algorithms

ESPRIT HSLA PROJECT

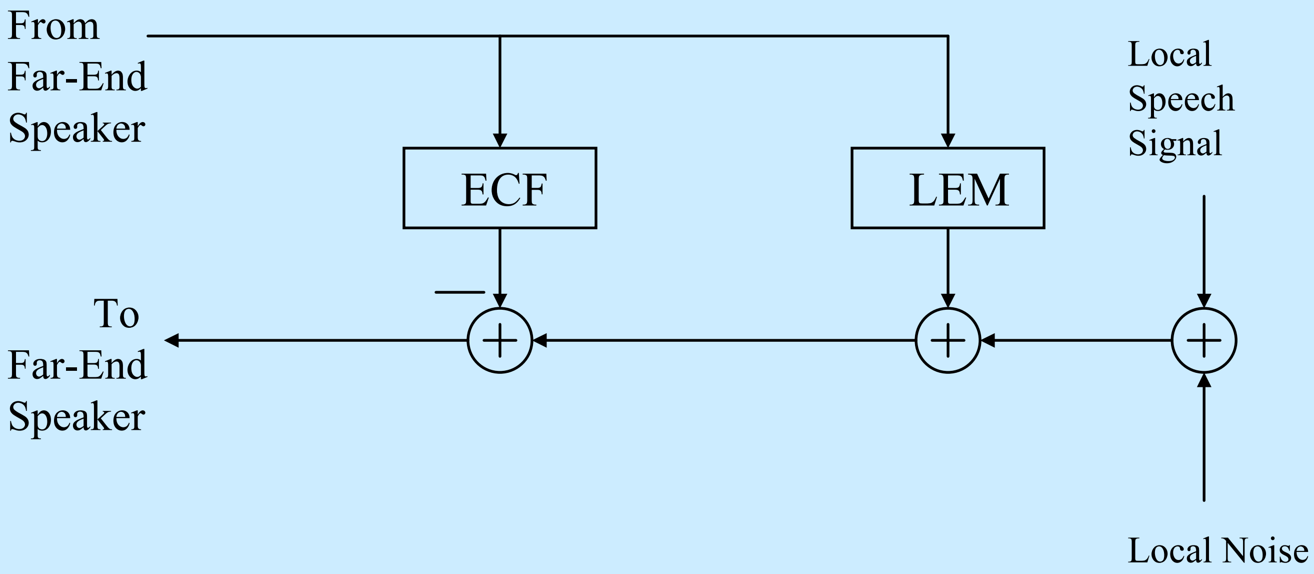
Felix Albu, Anthony Fagan UCD
Jiri Kadlec, Antonin Hermanek UTIA
Nick Coleman Univ. of Newcastle

Table of Contents

- Acoustic Echo Cancellation
- Logarithmic number system
- Fast Affine Projection (FAP) algorithms
- Conjugate Gradient Fast Affine Projection (CGFAP) Algorithm
- Simulations
- Conclusions

Acoustic echo cancellation

- Loudspeaker-enclosure-microphone (LEM) with an echo-cancellation filter (ECF)

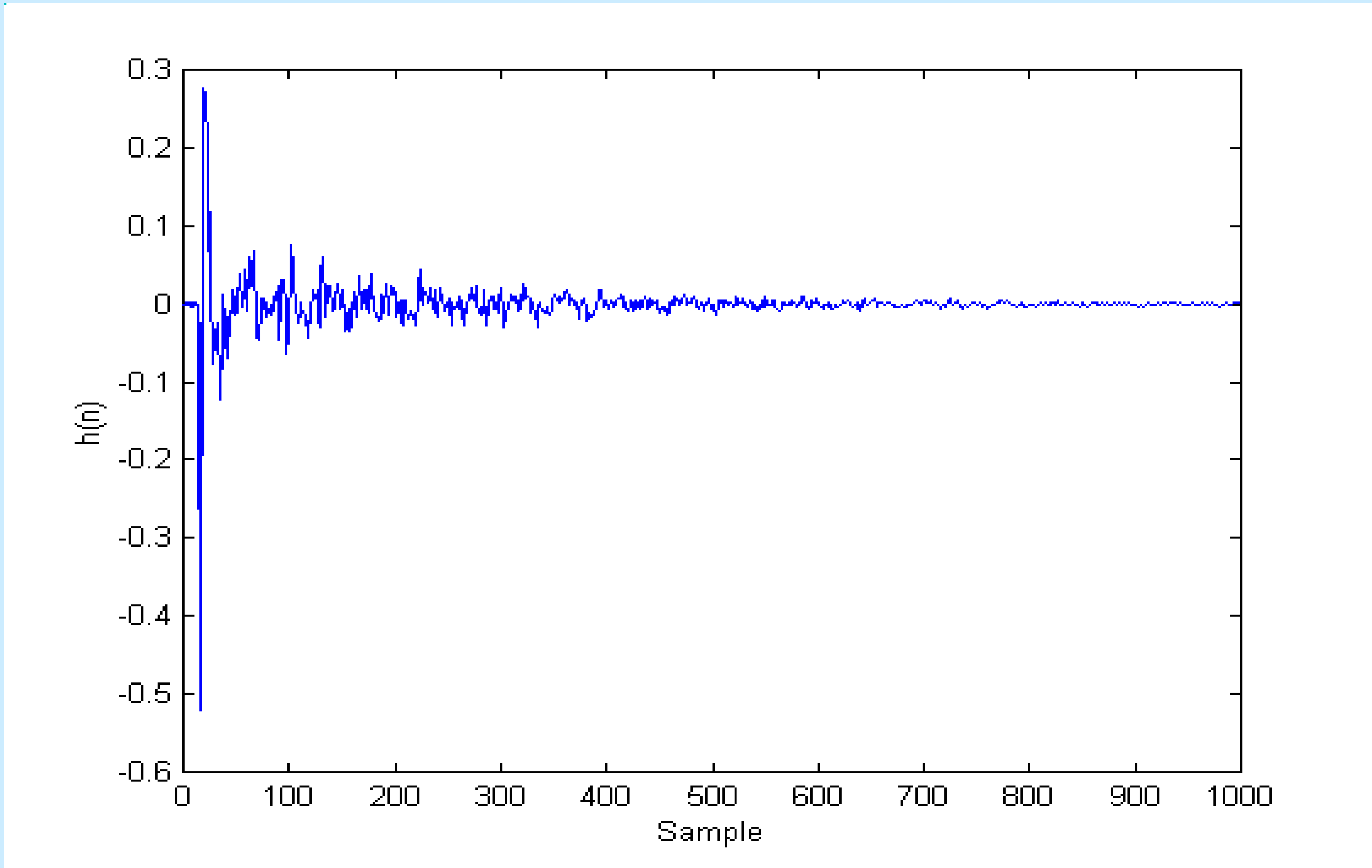


Acoustic echo cancellation

- The echo path is very long (~125 ms)
- The echo path may rapidly change at any time
- The impulse response varies with ambient temperature, pressure, humidity, movement of objects

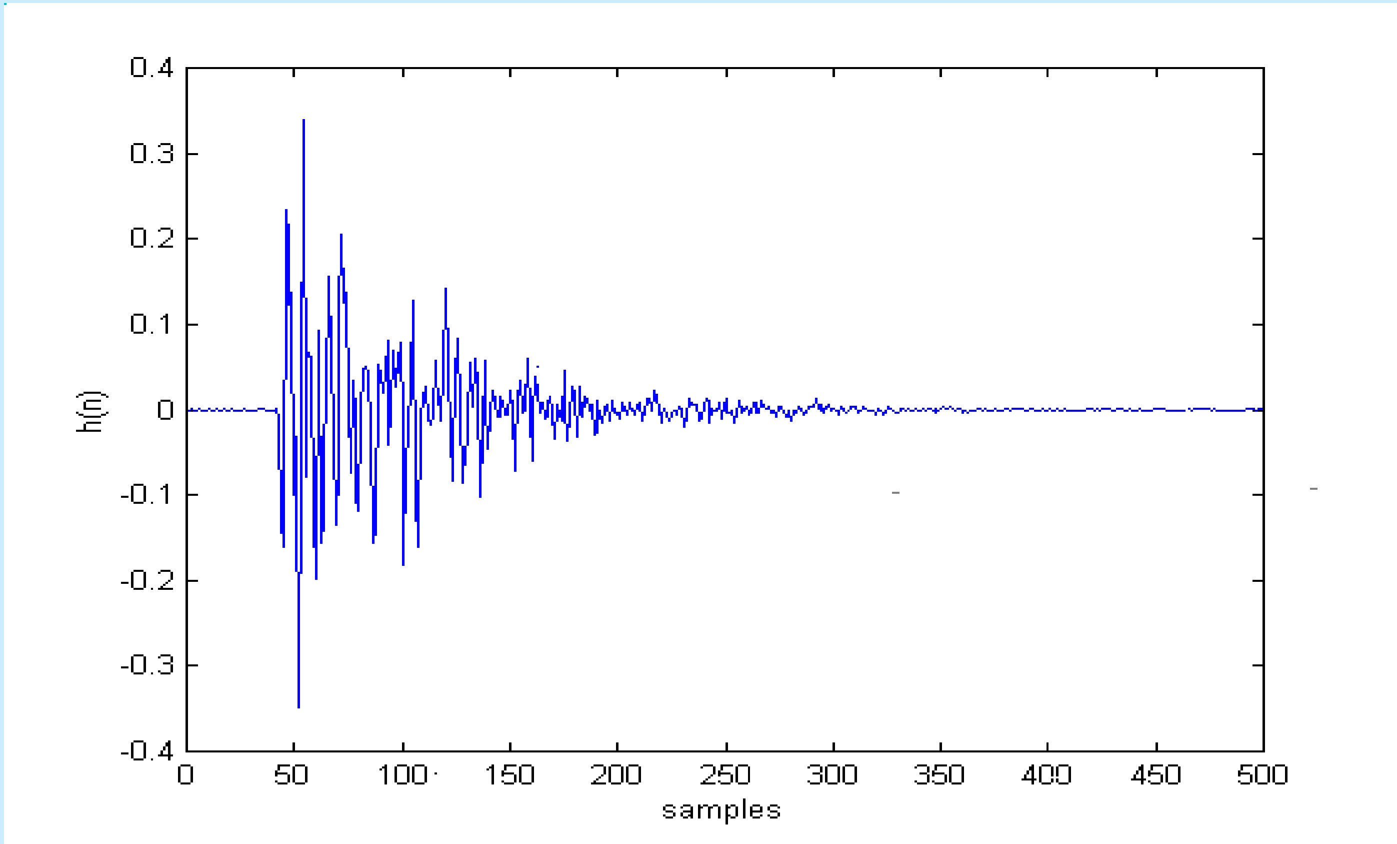
Acoustic echo cancellation

- The room impulse response



Acoustic echo cancellation

- The car impulse response



Logarithmic number system

IEEE single precision:

S	Exponent	Mantissa
31	30 … 23	22 … 0

32b LNS:

S	Exp integer	Exp fraction part
31	30 … 23	22 … 0

Elementary LNS operations:

x + y	ADD	Lz=Lx+log(1+2^(Ly-Lx)), Sz depends on sizes of x, and y
x - y	SUB	Lz=Lx+log(1- 2^(Ly-Lx)), Sz depends on sizes of x and y
x * y	MUL	Lz=Lx+Ly, Sz=Sx OR Sy
x / y	DIV	Lz =Lx-Ly, Sz=Sx OR Sy
x^0.5	SQRT	Lx >> 1, Sz=Sx

FAP Algorithms

Affine Projection Algorithm (APA) is a generalisation of the NLMS algorithm

1) $\underline{e}_n = \underline{s}_n - \mathbf{X}_n^t \underline{h}_{n-1}$

2) $\underline{\varepsilon}_n = \left[\mathbf{X}_n^t \mathbf{X}_n + \delta \mathbf{I}\right]^{-1} \underline{e}_n$

3) $\underline{h}_n = \underline{h}_{n-1} + \mu_A \mathbf{X}_\mathbf{n} \underline{\varepsilon}_n$

The complexity of APA is $2LN + O(N^2)$ where L is the length of the adaptive filter, N is the size of the projection .

FAP Algorithms

0) Initialization: $\underline{a}_0 = \left[1, \underline{0}^t\right]^t, \underline{b}_0 = \left[\underline{0}^t, 1\right]^t, \; E_{a,n} = E_{b,n} = \delta$	
1) Use sliding windowed FTF algorithm to update $E_{a,n}, E_{b,n}, \underline{a}_n$, and $\underline{b}_n$	10N
2) $\underline{\tilde{r}}_{xx,n} = \underline{\tilde{r}}_{xx,n-1} + x_n \underline{\tilde{\alpha}}_n - x_{n-L} \underline{\tilde{\alpha}}_{n-L}$	2N
3) $\hat{e}_n = s_n - \underline{x}_n^t \underline{\hat{h}}_{n-1}$	L
4) $e_n = \hat{e}_n - \mu \underline{\tilde{r}}_{xx,n}^t \overline{\underline{E}}_{n-1}$	N
5) $\underline{e} = \begin{bmatrix} e_n \\ (1-\mu)\overline{\underline{e}}_{n-1} \end{bmatrix}$	N
6) $\underline{\varepsilon} = \begin{bmatrix} 0 \\ \underline{\tilde{\varepsilon}}_n \end{bmatrix} + \frac{1}{E_{a,n}} \underline{a}_n \underline{a}_n^t \underline{e}$	2N
7) $\begin{bmatrix} \overline{\underline{\varepsilon}}_n \\ 0 \end{bmatrix} = \underline{\varepsilon}_n - \frac{1}{E_{b,n}} \underline{b}_n \underline{b}_n^t e_n$	2N
8) $\underline{E}_n = \begin{bmatrix} 0 \\ \overline{\underline{E}}_{n-1} \end{bmatrix} + \underline{\varepsilon}_n$	N
9) $\underline{\hat{h}}_n = \underline{\hat{h}}_{n-1} + \mu \underline{x}_{n-(N-1)} E_{N-1,n}$	L
10) $\underline{\hat{\varepsilon}}_{n+1} = (1-\mu)\overline{\underline{\varepsilon}}_n$	N

Total: $2L + 20N$

CGFAP Algorithm

Initialisation **(Conjugate Gradient FAP algorithm)**

$$0.\ \underline{V}(-1) = \underline{0},\ \underline{\eta}(-1) = 0,\ \underline{s}(-1) = 0,\ \mathbf{R}(-1) = \delta\mathbf{I},\ \alpha = 1,\ \underline{P}(-1) = \underline{b}/\delta$$

Processing in sampling interval *n*

1) $\mathbf{R}(n) = \mathbf{R}(n-1) + \underline{\xi}(n)\underline{\xi}^T(n) - \underline{\xi}(n-L)\underline{\xi}^T(n-L)$

2) $\underline{g}(n) = \mathbf{R}(n)\underline{P}(N-1) - \underline{b}$

3) $\gamma(n) = \dfrac{\underline{g}^T(n)\mathbf{R}(n-1)\underline{s}(n-1)}{\underline{s}^T(n-1)\mathbf{R}(n-1)\underline{s}(n-1)}$

4) $\underline{s}(n) = \gamma(n)\underline{s}(n-1) - \underline{g}(n)$

5) $\underline{P}(n) = \underline{P}(n-1) - \dfrac{\underline{g}^T(n)\underline{s}(n)}{\underline{s}^T(n)\mathbf{R}(n)\underline{s}(n)}\underline{s}(n)$

6) $\underline{V}(n) = \underline{V}(n-1) + \alpha\eta_{N-1}(N-1)\underline{X}(n-N)$

7) $y(n) = \underline{V}^T(n)\underline{X}(n) + \alpha\underline{\overline{\eta}}^T(n-1)\underline{\tilde{R}}(n)$

8) $e(n) = d(n) - y(n)$

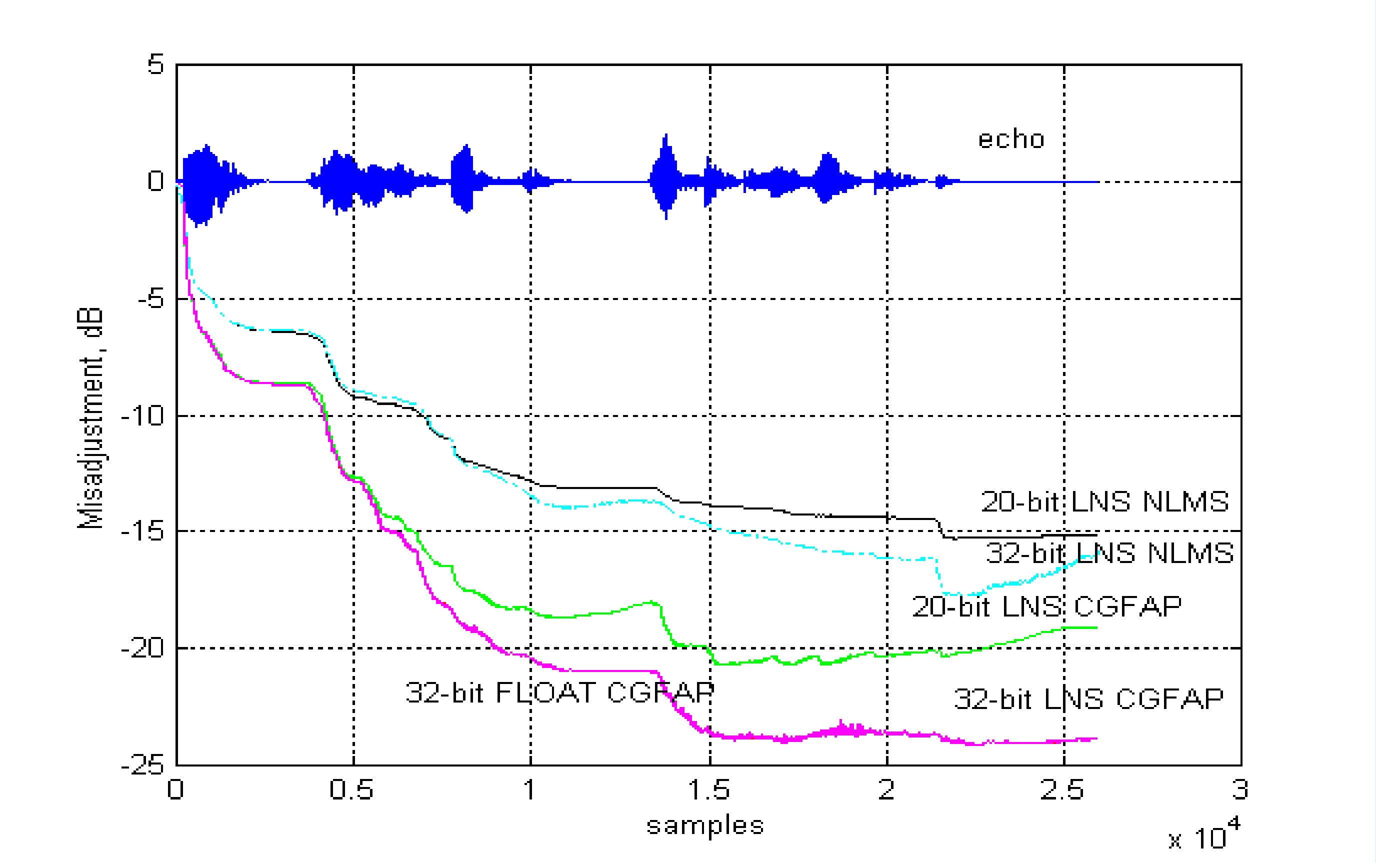
9) $\underline{\varepsilon} = e(n)\underline{P}(n)$

10) $\underline{\eta}(n) = \begin{bmatrix} 0 \\ \underline{\overline{\eta}}(n-1) \end{bmatrix} + \underline{\varepsilon}(n)$

Total : $2L + 2N^2 + 9N + 1$ (1 division)

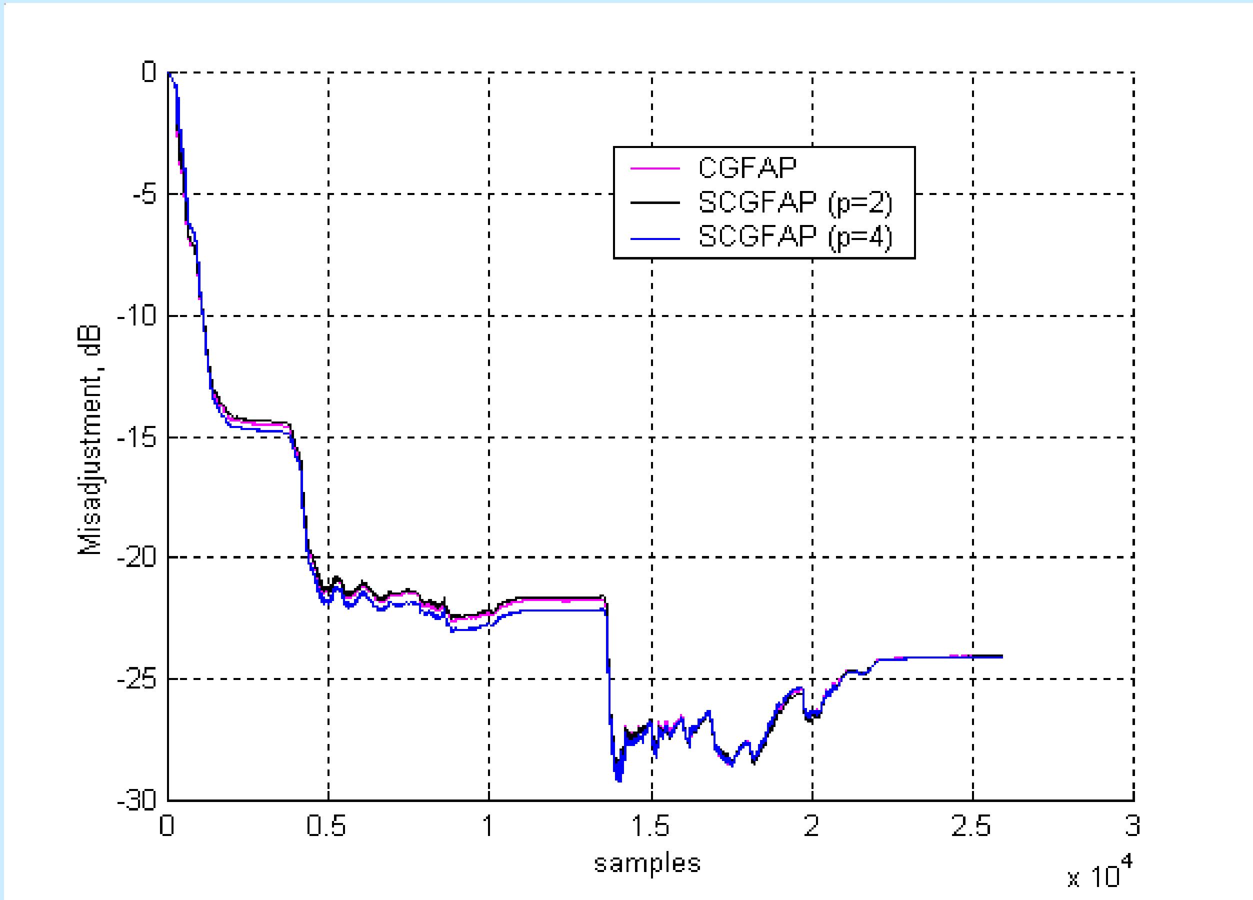
Simulations

The learning curves for 32-bit FLOAT, 32-bit and 20-bit LNS implementations of CGFAP algorithm (32-bit curves almost co-incidental) and DOUBLE NLMS algorithm (L=1000, N=10)



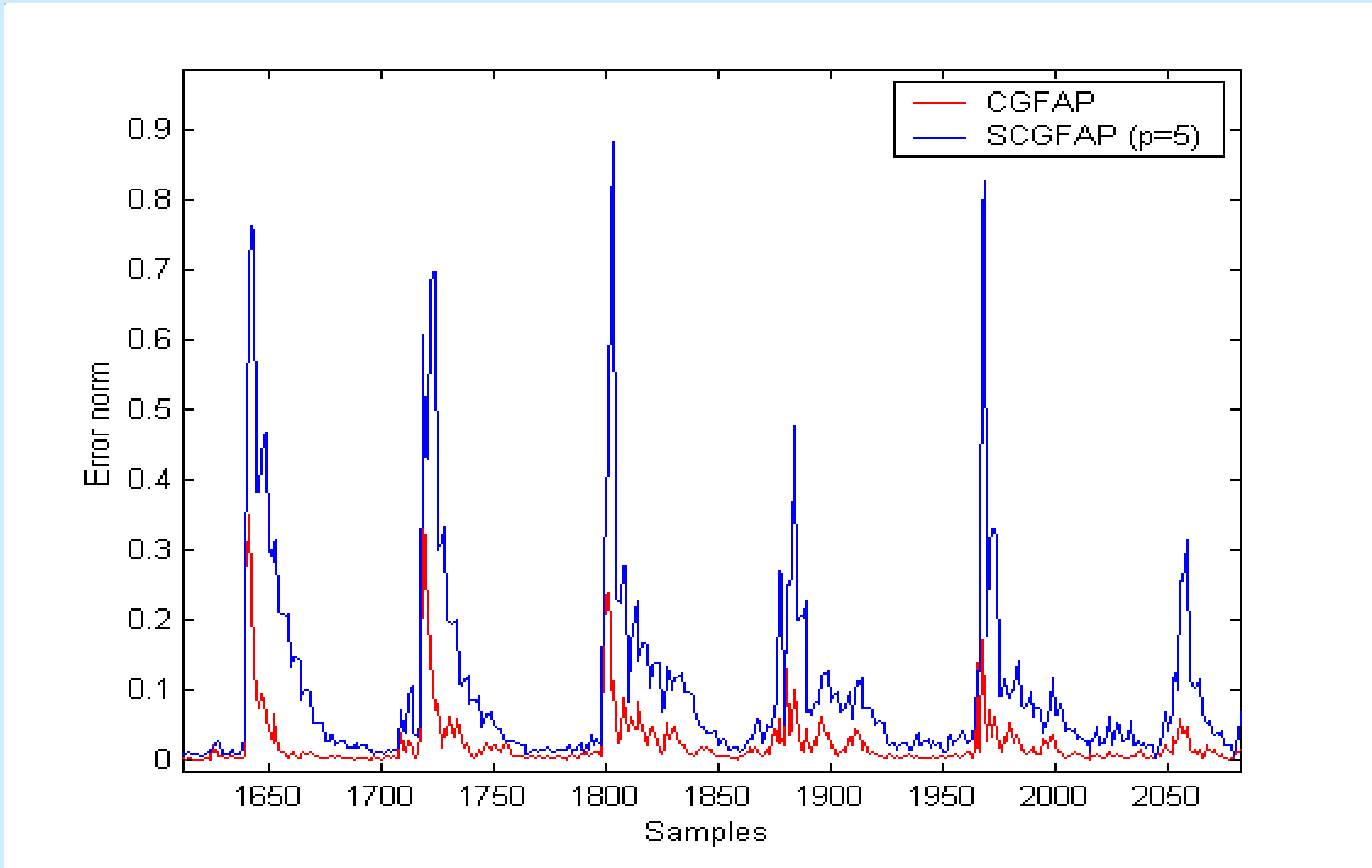
Simulations

Convergence of 20-bit LNS CGFAP implementation for different values of *p* (L=256, N=10)



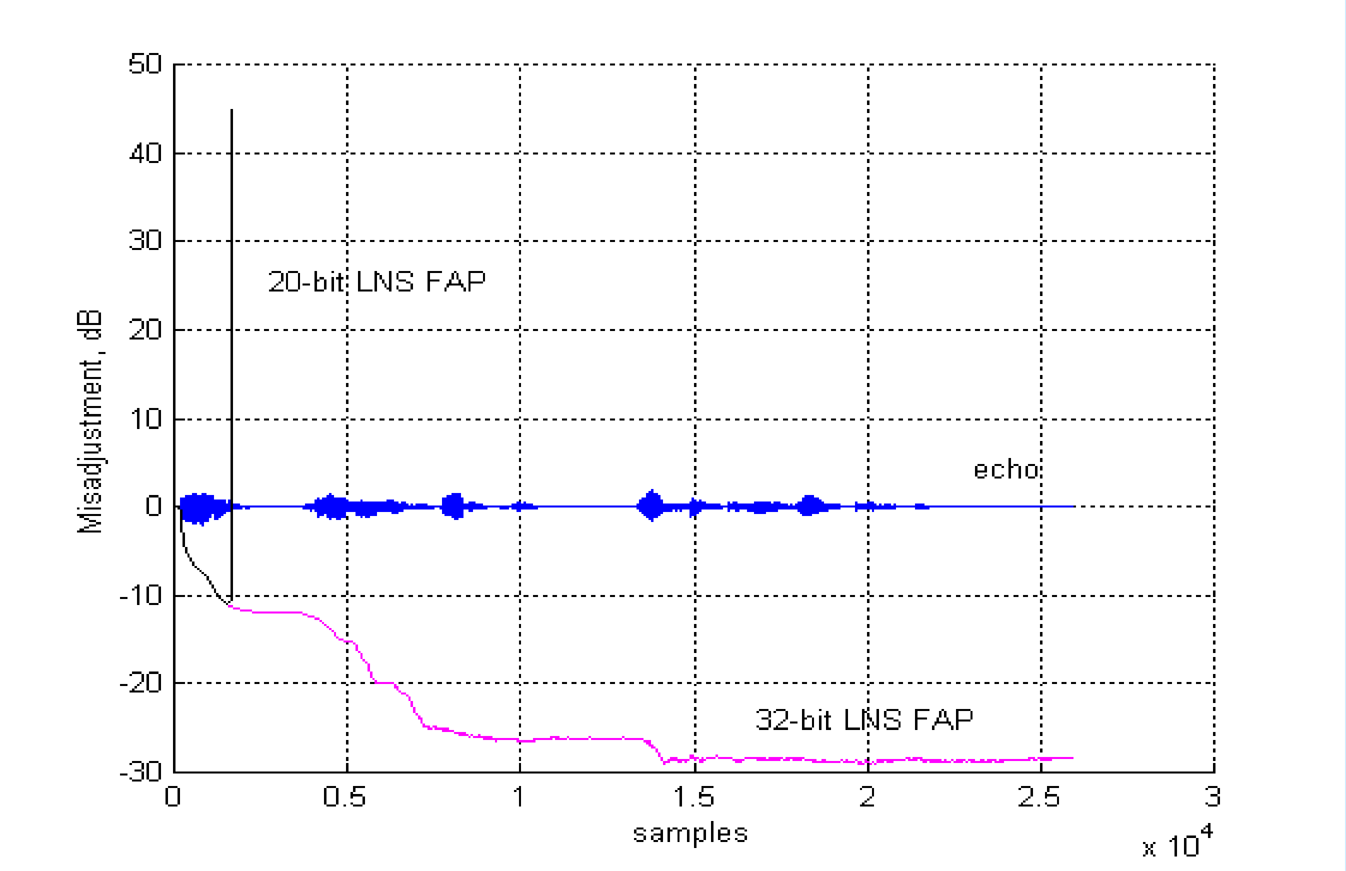
Simulations

The error norm between the exact solution (double precision) and the iterated solution of the linear system for different values of p (p=1 and p=5)



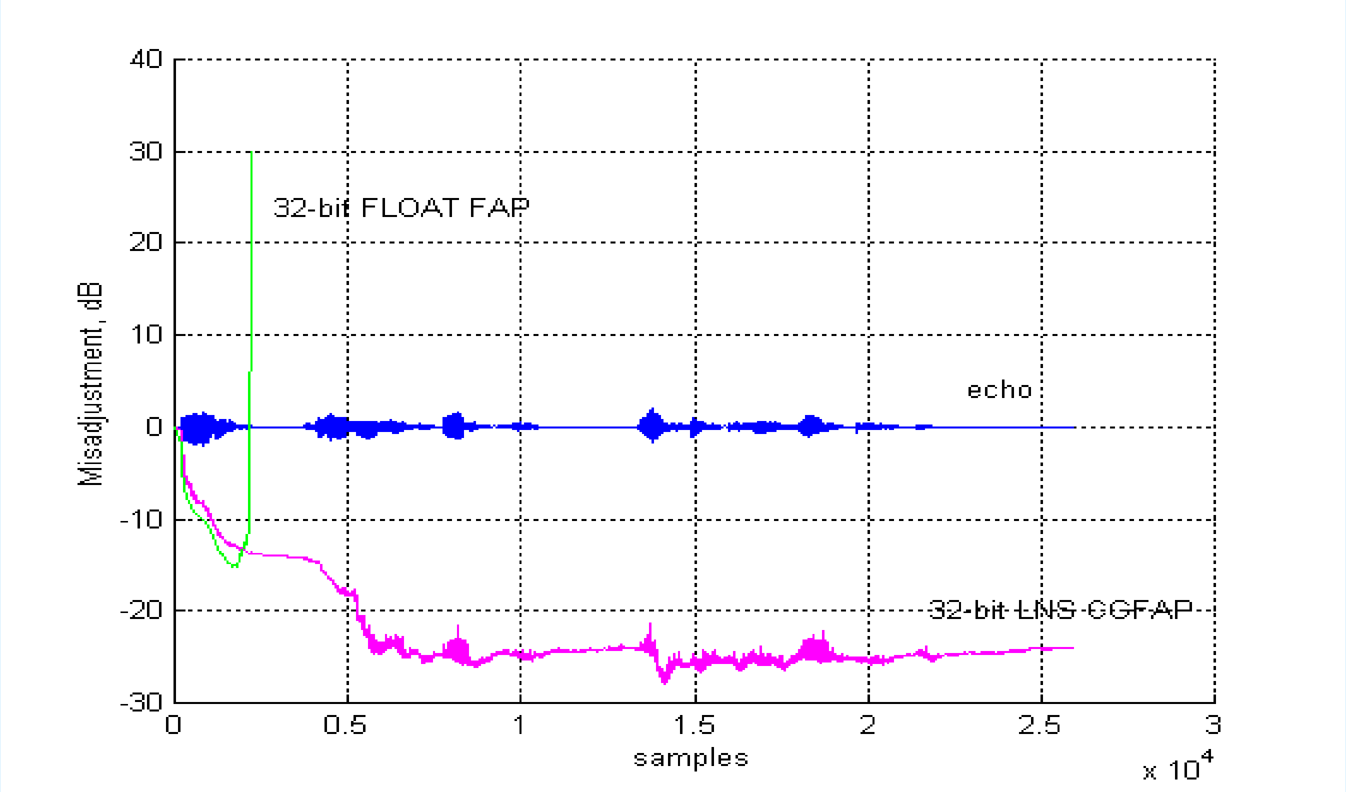
Simulations

Convergence of 32-bit LNS FAP implementation versus 20-bit FLOAT FAP implementation, Float is unstable after about 1600 iterations (L=256, N=10, k=100)



Simulations

- Convergence of 32-bit LNS CGFAP implementation versus 32-bit FLOAT FAP implementation, Float is unstable after about 2200 iterations (L=256, N=10, k=5)



Simulations

We can update $\underline{P}(n)$ less frequently without affecting too much the output error. Therefore, the average number of MACs is

$$2L+2N^2/p+(4+5/p)N-1+2/p$$

If L=1000 and N=10, NLMS needs 2025 MACs (assuming 25 MACs for a division)

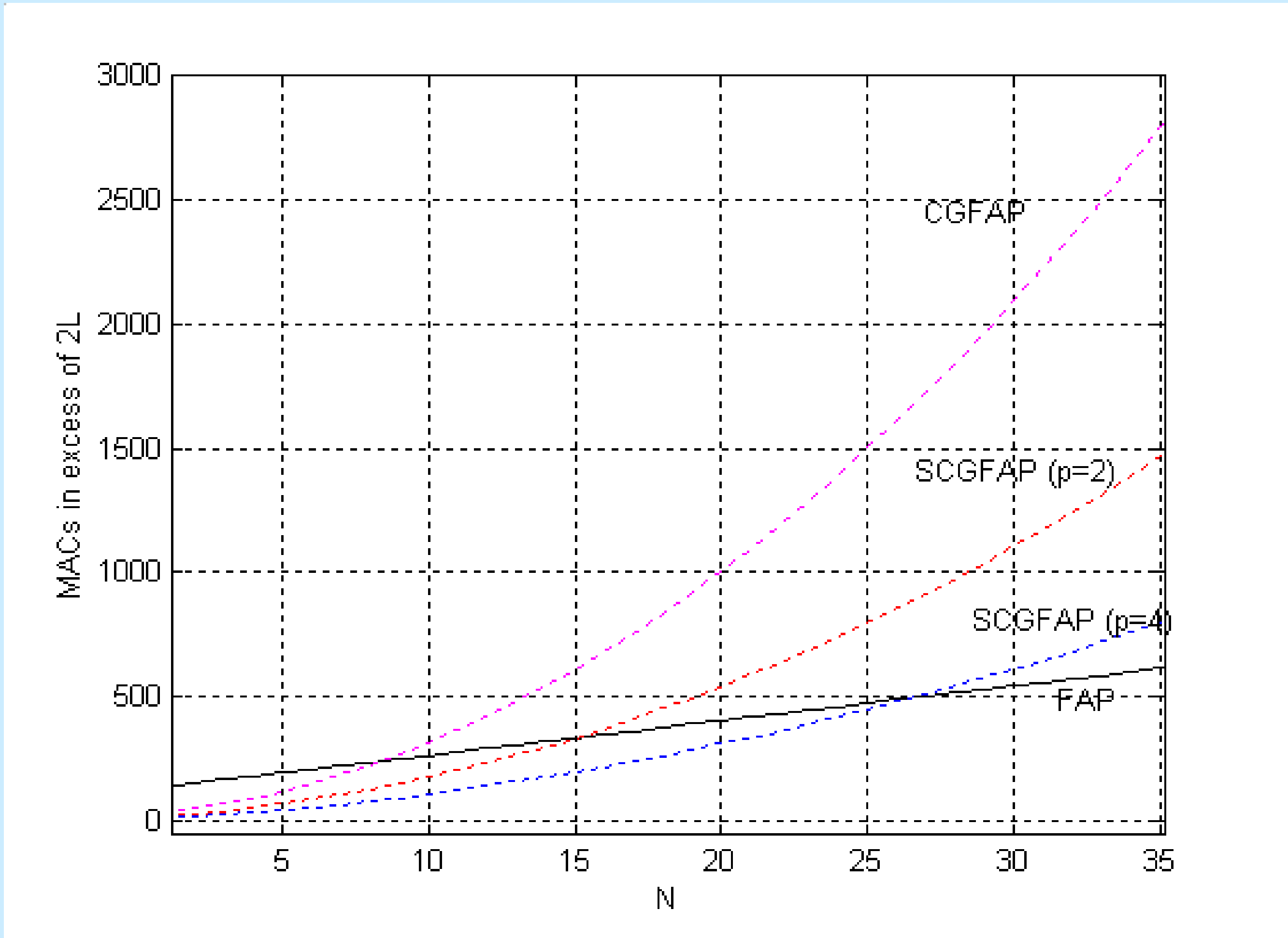
-FAP needs 2265 FAPs ($2L+20N$, 5 divisions)

- CGFAP needs 2316 MACs ($2L+2N^2+9N+1$, 1 division)

- SCGFAP needs 2108 MACs ($2L+2N^2/p+(4+5/p)N-1+2/p$, p=4)

Simulations

Real time requirements of 3 Fast Affine Projection algorithms



Conclusions

- [1] J.N. Coleman, E.I.Chester, 'A 32-bit Logarithmic Arithmetic Unit and Its Performance Compared to Floating-Point', *14th Symposium on Computer Arithmetic',* Adelaide, April 1999
- [2] C. Breining, P. Dreitseitel, E. Hansler, A. Mader, B. Nitsch, H. Pudeer, T. Scheirtler, G. Schmidt, and J.Tilp, ' Acoustic echo control- An application of very high order adaptive filters,' *IEEE Signal Processing Magazine*, pp. 42-69, July 1999
- [3] K. Ozeki, T. Umeda, 'An adaptive Filtering Algorithm Using an Orthogonal Projection to an Affine Subspace and its Properties,' Electronics and Communications in Japan, Vol. 67-A, No.5, 1984
- [4] S. Gay, S. Tavathia, 'The Fast Affine Projection Algorithm', pp. 3023–3026, *ICASSP'95 Proceedings*
- [5] S. Gay, J. Benesty, editors, 'Acoustic Signal Processing for Telecommunication', Kluwer Academic Publishers, 2000
- [6] Y. Kaneda, M. Tanaka, J. Kojima, 'An Adaptive Algorithm with Fast Convergence for Multi-input Sound Control', Active95, pp. 993-1004, Newport Beach, California, USA
- [7] Q.G. Liu, B. Champagne, and K. C. Ho, " On the use of a modified FAP algorithm in subbands for acoustic echo cancellation," in Proc. 7th IEEE DSP Workshop, Loen, Norway, 1996, pp. 2570-2573
- [8] M. Ghanassi, B. Champagne, "On the Fixed-Point Implementation of a Subband Acoustic Echo Canceler Based on a Modified FAP Algorithm", 1999 IEEE Workshop on Acoustic Echo and Noise Control, Pocono Manor, Pennsylvania, USA pp. 128-131
- [9] Heping Ding, "A stable fast affine projection adaptation algorithm suitable for low-cost processors", ICAASP 2000, Turkey, pp. 360-363
- [10] David Luenberger, "Linear and Non-linear Programming", 2nd Edition, Addison-Wesley, 1984
- [11] J.N.Coleman, E.Chester, C.Softley and J.Kadlec, "Arithmetic on the European Logarithmic Microprocessor", IEEE Trans. Comput. Special
- Edition on Computer Arithmetic, July 2000, vol. 49, no. 7, pp. 702-715; and erratum October 2000, vol. 49, no. 10, p.1152.
- [12] Erwin Kreyszig, 'Advanced Engineering mathematics', 7th edition, John Wiley & Sons, 1993
- [13] R.Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. van der Vorst, 'Templates for the
- solutions of linear systems: Building blocks for iterative methods', SIAM, 1994
- [14] F. Albu, J. Kadlec, C. Softley, R. Matousek, A. Hermanek, N. Coleman, A. Fagan, "Implementation of (Normalised) RLS Lattice on Virtex", FPL2001, pp. 91-100, Belfast, UK.

Conclusions

- The SCGFAP Algorithm is a stable FAP algorithm. It is only marginally complex than NLMS, but achieves substantial improvements.
- Its 32-bit and 20-bit LNS are easy to implement. Also, it is suitable to implement with most commercial DSPs because of its reduced memory requirements and low complexity (just 1 division).
- SCGFAP algorithm is a good candidate for different voice applications.

Questions ?

- HSLA project website
  http://napier.ncl.ac.uk/hsla
- UCD's DSP Group website
  http://dsp.ucd.ie