Self-Similarity of A Reverse-Filter Generator

Jin-Yuan Chen

Advisor: Prof. Po-Ning Chen

Department of Communications Engineering
National Chiao Tung University, Taiwan 30056, R.O.C.

July, 2003

OUTLINE

- Introduction
- Self-Similar Process and Analysis Tests
- Schemes of Self-Similar Traffic Generator
- Simulation of Traffic Traces
- Conclusion

Introduction

Preliminaries

- Recent measurement suggested that a good network traffic model should emulate the *long-range dependence*(LRD) or *self-similarity* nature of real network traffic.
- Incorrect assessments of network system may be obtained, if LRD nature is not considered for the experimental synthetic network traffic.
- In this thesis, we follow the work of Hua and Chen by replacing the forward filter by an equivalent reverse filter.
- The key difference between forward- and reverse-filter traffic generators is that the forward method uses an FIR (finite impulse response) filter, while the reverse method employs an IIR (infinite impulse response) filter.

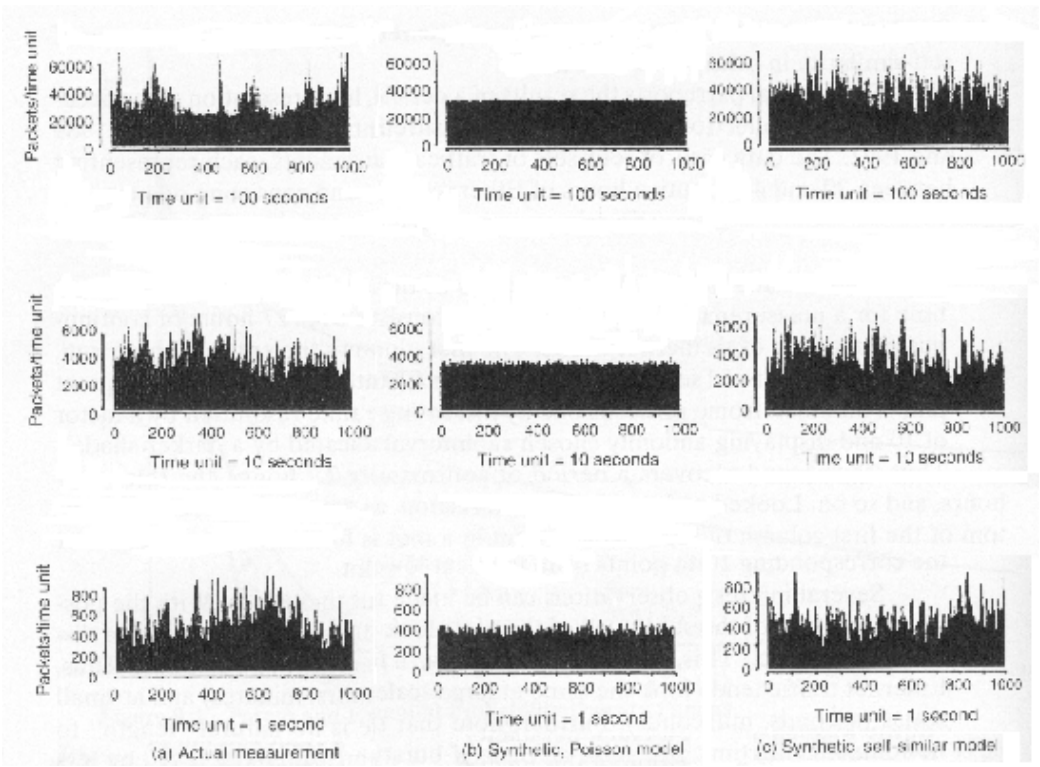


Illustration of actual Ethernet traffic and synthetic Ethernet traffic of Poisson mode and self-similar mode.

(Source: Fig. 9.5 in *High-Speed Networks and Internets* by Stalling, 2002)

Self-Similar Process and Analysis Tests

Long-Range Dependence

Long-range dependence is defined in terms of the behavior of the autocovariance function $B(k)$ of a stationary process.

Definition 1 *A discrete-time second-order-stationary process $X_1, X_2, \ldots$ is said to be* long-range dependent, *if*

$$\sum_{k=-\infty}^{\infty} |B(k)| = \infty.$$

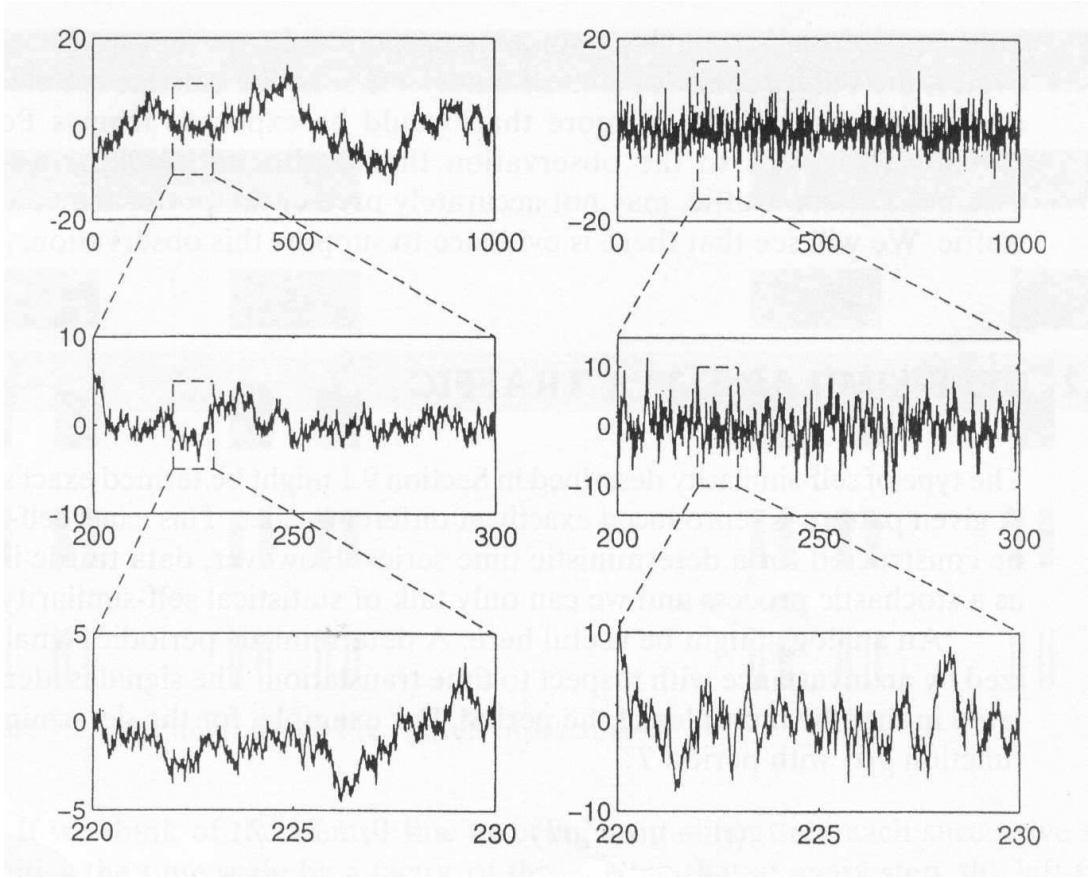
Definition 2 *A discrete-time second-order-stationary process $X_1, X_2, \ldots$ is said to be* short-range dependent, *if*

$$\sum_{k=-\infty}^{\infty} |B(k)| < \infty.$$

Definition 3 *A discrete-time second-order-stationary process $X_1, X_2, \ldots$ is said to be* long-range dependent, *$0 < \beta < 1$, if*

$$B(k)/B(0) \sim k^{-\beta} \quad \text{as } k \to \infty,$$

Self-Similar Process



Compare the difference between self-similar process and non-self-similar process.

Self-Similar Process

Self-similarity

- **m-averaged process** $\boldsymbol{X}^{(m)} = (X_1^{(m)}, X_2^{(m)}, \ldots)$ of a discrete-time stationary parent process $X_1, X_2, \ldots$ as:

$$X_i^{(m)} = \frac{1}{m}\sum_{j=1}^{m} X_{m(i-1)+j} = \frac{X_{m(i-1)+1} + X_{m(i-1)+2} + \cdots + X_{mi}}{m}$$

- **Autocovariance function** of the m-averaged process $\boldsymbol{X}^{(m)}$

$$B_m(k) = \mathrm{Cov}\left\{X_i^{(m)}, X_{i+k}^{(m)}\right\}$$

Self-Similar Process

Self-similarity

- **Autocorrelation coefficient function** of the m-averaged process $\boldsymbol{X}^{(m)}$

$$R_m(k) = B_m(k)/B_m(0)$$

Definition 4 *A discrete-time second-order-stationary process* $\boldsymbol{X} = (X_1, X_2, X_3, \ldots)$ *is called* exactly second-order self-similar *with parameter* $H = 1 - (\beta/2)$, *where* $0 < \beta < 1$, *if its autocorrelation coefficient function is*

$$R_1(k) = \frac{1}{2}[|k+1|^{2H} - 2|k|^{2H} + |k-1|^{2H}] \text{ for } k = 1, 2, \ldots.$$

Self-Similar Process

Theorem

Theorem 1 *For a second-order-stationary process* $\boldsymbol{X} = (X_1, X_2, X_3, \ldots)$ *and* $0 < \beta < 1$, *the following statements are equivalent:*

1. $R_1(k) = (1/2)[|k+1|^{2-\beta} - 2|k|^{2-\beta} + |k-1|^{2-\beta}]$ *for* $k = 1, 2, \ldots$;
2. $B_m(0) = B_1(0)m^{-\beta}$ *for* $m = 1, 2, \ldots$;
3. $B_m(k) = B_1(k)m^{-\beta}$ *for* $m = 1, 2, \ldots$ *and* $k = 0, 1, 2, \ldots$.

- Theorem 1 indicates that the exactly second-order self-similarity can be equivalently defined by using any of three statements. Definition 4 just conveniently takes the first statement in its definition.

Self-Similar Process

Self-similarity

Using Taylor expansion, we can obtain that for $0.5 < H < 1$,

$$\begin{aligned} R_1(k) &= \frac{1}{2}\left[|k+1|^{2H} - 2|k|^{2H} + |k-1|^{2H}\right] \\ &= \frac{1}{2}k^{2H}\left[\left|1+k^{-1}\right|^{2H} - 2 + \left|1-k^{-1}\right|^{2H}\right] \\ &= \frac{1}{2}k^{2H}\big[\left(1 + 2Hk^{-1} + H(2H-1)k^{-2} + O(k^{-3})\right) \\ &\quad -2 + \left(1 - 2Hk^{-1} + H(2H-1)k^{-2} + O(k^{-3})\right)\big] \text{ as } k \to \infty \\ &= H(2H-1)k^{2H-2} + O(k^{2H-3}) \text{ as } k \to \infty. \end{aligned}$$

Therefore, an *exactly second-order self-similar* process is long-range dependent in the sense of Definition 3.

Slowly Decay Variance and Variance-Time Analysis

- From Theorem 1 and $H = 1 - (\beta/2)$,

$$\mathrm{Var}\{X^{(m)}\} = B_m(0) = B_1(0)m^{-\beta} = \frac{\mathrm{Var}\{X\}}{m^{2-2H}}.$$

- The variance of m-averaged process $\boldsymbol{X}^{(m)}$ decreases more slowly than the reciprocal of the average size m for an exactly second-order self-similar process $\boldsymbol{X}$ with $0.5 < H < 1$.
- In fact, $\left(\frac{\mathrm{Var}\{X^{(m)}\}}{\mathrm{Var}\{X\}}\right)$ decreases as a slope of $(2H-2)$ in log-log plot against m.
- Thus, if we plot the curve of $\log(\mathrm{Var}\{X^{(m)}\}/\mathrm{Var}\{X\})$ against $\log(m)$, we will get a straight line with slope $(2H-2)$, which can be used to determine H.

Schemes of Self-Similar Traffic Generator

Existing Techniques for the Generation of Self-Similar Traffics

- Fast Fourier Transform Traffic Synthesizer (Paxson 1995)
- A Linear Approximation to FFT Traffic Synthesizer (Ledesma and Liu 2000)
- Forward-Filter-Based Self-Similar Traffic Generator (Hua and Chen 2002)

Fast Fourier Transform Traffic Synthesizer

- The method uses the Discrete-Time Fourier Transform (DTFT) to synthesize the fractional Gaussian noise (FGN).
- Sampling the power spectrum of the FGN, and taking inverse-DTFT of these samples, the time-domain self-similar arrivals are readily obtained.
- The power spectrum of the FGN process

$$S_y(w) = 2\,\sin(\pi H)\,\Gamma(2H+1)\,\left[1-\cos(w)\right]\,\left[\,|w|^{-2H-1} + A(w)\right]$$

where $A(w) = \sum_{k=1}^{\infty}[(2\pi k + w)^{-2H-1} + (2\pi k - w)^{-2H-1}]$.

Fast Fourier Transform Traffic Synthesizer

The main difficulty of using above formula to obtain the samples of the power spectrum is the vexing infinite number of summands in $A(w)$. Paxson resolved the problem by approximating $A(w)$ as:

$$\begin{aligned} A(w) &\approx \sum_{k=1}^{3} \left[(2\pi k + w)^{-2H-1} + (2\pi k - w)^{-2H-1}\right] \\ &\quad + \frac{1}{8H\pi} \sum_{k=3}^{4} \left[(2\pi k + w)^{-2H-1} + (2\pi k - w)^{-2H-1}\right] \end{aligned}$$

Fast Fourier Transform Traffic Synthesizer

Paxson's method proceeds as follows:

- Construct a sequence of values S_1,..... $S_{n/2}$, where $S_j = \hat{S}_y(2\pi j/n)$.
- Multiply each sample by an independent exponentially distributed random variable with mean 1.
- Take the square root of all samples. Multiply each by an independent phase that is uniformly distributed over $[0, 2\pi)$.
- Make copies of the current samples to the other half of the spectrum (i.e., from $-\pi$ to 0) to ensure *symmetry*.
- Take Inverse Fast Fourier Transform to the $2n$ spectrum samples to obtain the approximate FGN sample path.

Fast Fourier Transform Traffic Synthesizer

- The Paxson's method can synthesize a self-similar process path with a moderate computational complexity.
- The traffic sequence cannot be generated on the fly.
- The resultant H may be slightly different from targeted H.
- The output sequence may be negative.

A Linear Approximation to FFT Traffic Synthesizer

Ledesma and Liu demonstrated that a linear approximation reduces the complexity of the computation without compromising the accuracy in synthesizing the power spectrum of the FGN.

Re-write $A(w)$ as:

$$\sum_{k=1}^{2}[(2\pi k + w)^{-2H-1} + (2\pi k - w)^{-2H-1}] + A_{3:\infty}$$

where

$$A_{3:\infty} = \sum_{k=3}^{\infty}[(2\pi k + w)^{-2H-1} + (2\pi k - w)^{-2H-1}]$$

Using a linear function $D(w) = p \cdot w + q$ to approximate $A_{3:\infty}$, and determining the optimal p and q according to the mean-squared-error criterion.

A Linear Approximation to FFT Traffic Synthesizer

Obtain:

$$p^* = -\frac{6}{\pi}F + \frac{12}{\pi^3}G$$

$$q^* = \frac{4}{\pi}F - \frac{6}{\pi^2}G$$

$$F = \sum_{k=3}^{\infty}\left[\frac{(2\pi k-\pi)^{-2H} - (2\pi k+\pi)^{-2H}}{2H}\right]$$

$$G = \sum_{k=3}^{\infty}\left(\frac{(2\pi k)[(2\pi k+\pi)^{-2H} + (2\pi k-\pi)^{-2H} - 2(2\pi k)^{-2H}]}{2H} - \frac{[(2\pi k+\pi)^{-2H+1} + (2\pi k-\pi)^{-2H+1} - 2(2\pi k)^{-2H+1}]}{2H-1}\right)$$

A Linear Approximation to FFT Traffic Synthesizer

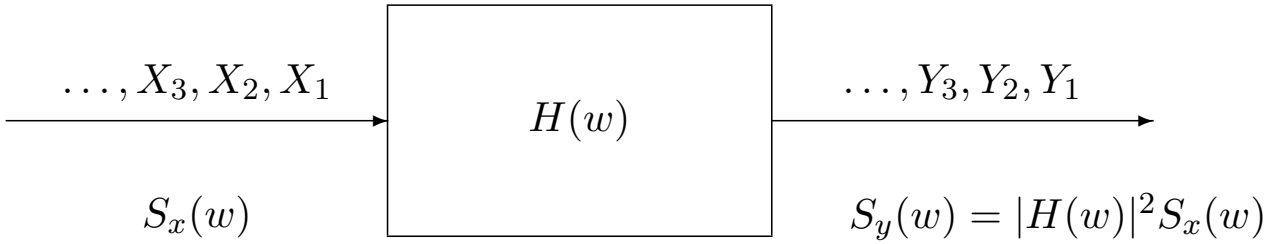
Length	Ledesma and Liu	Paxson
65,536	5	7
131,072	10	12
262,144	22	24
524,288	45	48
1,048,576	94	260
2,097,152	259	659

Execution time in seconds for synthesizing a self-similar sequence. (Source: Ledesma and Liu 2000)

A Linear Approximation to FFT Traffic Synthesizer

- The Paxson's method requires more computation time than the linear approximation method.
- Similar to Paxson's method, the linear approximation method requires the knowledge of the sequence size before its execution.
- The linear approximation method also cannot generate self-similar traffic on the fly.
- The output sequence may be negative.

Forward-Filter-Based Self-Similar Traffic Generator



- It is also designed based on power-spectrum fitting.
- Power spectrum of an exactly second-order self-similar process

$$S_y(w) = \sin(\pi H) \cdot \Gamma(2H+1) \cdot |1 - e^{-jw}|^2 \sum_{k=-\infty}^{\infty} |w + 2\pi k|^{-1-2H} \quad \text{for } -\pi \leq w < \pi.$$

Forward-Filter-Based Self-Similar Traffic Generator

- Similar to paxson's method, the forward-filter-based self-similar traffic generator approximates the above infinite summation by a finite sum.
- Hua and Chen proposed to only take the main term $k = 0$, and yields that

$$S_y(w) \approx \sin(\pi H) \cdot \Gamma(2H+1) \cdot |1 - e^{-jw}|^2 \cdot |w|^{-1-2H} \quad \text{for } -\pi \leq w < \pi.$$

- By observing that the degree of self-similarity is mostly determined by the spectrum behavior close to origin, they further approximate $|1 - e^{-jw}|$ by $|w|$.

$$\tilde{S}_y(w) = |1 - e^{-jw}|^{1-2H} \quad \text{for } -\pi \leq w < \pi,$$

where the coefficient, $\sin(\pi H) \cdot \Gamma(2H+1)$, is removed for simplicity.

Forward-Filter-Based Self-Similar Traffic Generator

$$
\begin{aligned}
|1-e^{-jw}| &= |1-\cos(w)+j\sin(w)| \\
&= \sqrt{[1-\cos(w)]^2+\sin^2(w)} \\
&= \sqrt{1-2\cos(w)+\cos^2(w)+\sin^2(w)} \\
&= \sqrt{2[1-\cos(w)]} \\
&= \sqrt{2\{[\cos^2(w/2)+\sin^2(w/2)]-[\cos^2(w/2)-\sin^2(w/2)]\}} \\
&= \sqrt{2[2\sin^2(w/2)]} \\
&= 2|\sin(w/2)| \\
&\cong 2|w/2| = |w|, \quad \text{as } |w| \text{ close to } 0.
\end{aligned}
$$

Forward-Filter-Based Self-Similar Traffic Generator

- It remains to design a filter whose output spectrum due to an i.i.d. input equals $\tilde{S}_y(w)$, or specifically, $|H(w)|^2 = |1 - e^{-jw}|^{1-2H}$.
- By Taylor's expansion,

$$(1-z)^{-(2H-1)/2} = \sum_{n=0}^{\infty} \frac{\Gamma(n+(2H-1)/2)}{\Gamma(n+1)\Gamma((2H-1)/2)} z^n,$$

  where $\Gamma(\cdot)$ represents the Euler gamma function.

- Replacing z with e^{-jw} gives that

$$(1-e^{-jw})^{(1-2H)/2} = \sum_{n=0}^{\infty} \frac{\Gamma(n+H-0.5)}{\Gamma(n+1)\Gamma(H-0.5)} e^{-jwn}.$$

Forward-Filter-Based Self-Similar Traffic Generator

- Term-wisely comparing with $H(w) = \sum_{n=0}^{\infty} h_f[n]e^{-jwn}$ concludes that

$$h_f[n] = \begin{cases} \dfrac{\Gamma(n + H - 0.5)}{\Gamma(n+1)\Gamma(H-0.5)}, & \text{for } n \geq 0 \\ \\ 0, & \text{otherwise,} \end{cases}$$

- The relation between input sequence $x[n]$ and output sequence $y[n]$ is equal to:

$$y[n] = \sum_{k=0}^{\infty} h_f[k] \cdot x[n-k].$$

Reverse-Filter-Based Self-Similar Traffic Generator

From the design of the forward-filter-based self-similar traffic generator, we learn that the z-transforms of input $X(z)$ and output $Y(z)$ can be characterized by:

$$H(z) = (1 - z^{-1})^{(1-2H)/2} = \frac{1}{(1 - z^{-1})^{(2H-1)/2}} = \frac{Y(z)}{X(z)}$$

$$(1 - z^{-1})^a \, Y(z) = X(z)$$

where $a = (2H - 1)/2 \in (0, 0.5)$.

The binomial expansion of $(1 - z^{-1})^a$ is equal to:

$$\begin{aligned}(1 - z^{-1})^a &= 1 + \frac{-a}{1!}z^{-1} + \frac{-a(1-a)}{2!}z^{-2} + \cdots + \frac{-a(1-a)\cdots(n+1-a)}{n!}z^{-n} + \cdots \\ &= 1 - a\sum_{n=1}^{\infty} \frac{\Gamma(n-a)}{\Gamma(n+1)\Gamma(1-a)}z^{-n}\end{aligned}$$

Reverse-Filter-Based Self-Similar Traffic Generator

This new expression shows that the outputs $y[1], y[2], y[3] \ldots$ can also be obtained through:

$$\begin{aligned} y[n] &= x[n] + a \sum_{k=1}^{\infty} \frac{\Gamma(k-a)}{\Gamma(k+1)\Gamma(1-a)} y[n-k] \\ &= x[n] + \sum_{k=1}^{\infty} h_r[k] \cdot y[n-k] \end{aligned}$$

where

$$\begin{aligned} h_r[k] &= \begin{cases} \dfrac{a \cdot \Gamma(k-a)}{\Gamma(k+1)\Gamma(1-a)}, & \text{for } k \geq 1 \\ 0, & \text{otherwise,} \end{cases} \\ &= \begin{cases} \dfrac{(H-0.5) \cdot \Gamma(k-H+0.5)}{\Gamma(1.5-H)\Gamma(k+1)}, & \text{for } k \geq 1 \\ 0, & \text{otherwise,} \end{cases} \end{aligned}$$

Reverse-Filter-Based Self-Similar Traffic Generator

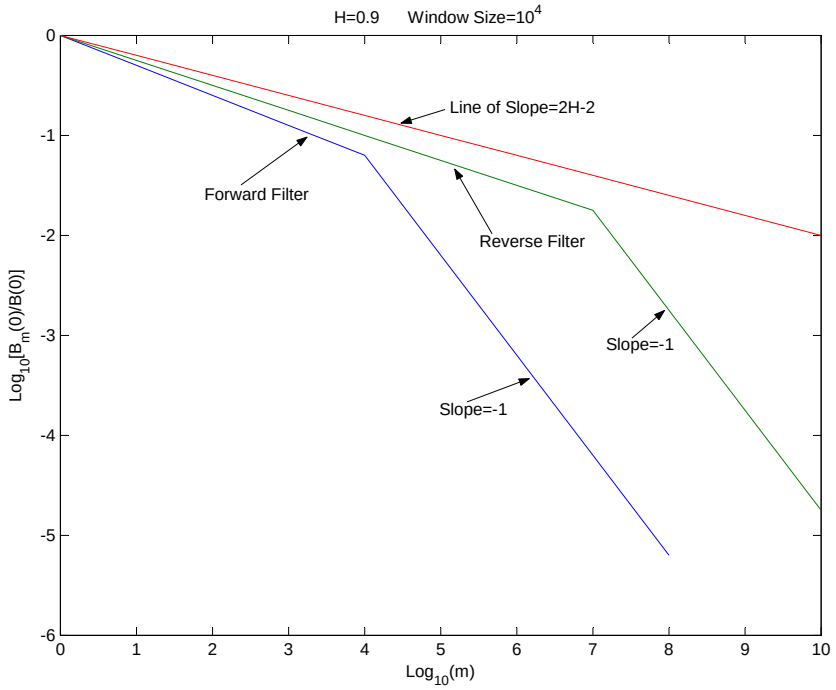
The key difference between forward- and reverse-filter traffic generators:

- The forward method uses an FIR (finite impulse response) filter.
- The reverse method employs an IIR (infinite impulse response) filter.
- Unlike the forward filter model, the above formula gives an infinite impulse response filter (IIR) even if a finite truncation on $h_r[\cdot]$ is applied.
- Notably, without truncation on the filter, the reverse filter model is indeed equivalent to the forward filter model used by Hua and Chen.

Reverse-Filter-Based Self-Similar Traffic Generator

Again, the output of the reverse filter should have infinite memory, even with a finite truncation.

It is based on this reason that we conjectured the reverse filter approach with truncation should be more self-similar than the forward filter approach with truncation



Truncation Analysis and Integerization

- The forward filter and reverse filter (in their original un-truncated forms) are of infeasibly infinite length, and must be **truncated** (with truncation window W).
- **Integerization**
  - The output $y[n]$ of the (either forward or reverse) filter is in general not an integer.
  - They cannot be used as a network arrival sequence if no certain "rounding" mechanism is provided.
  - Hua and Chen proposed to use "rounding $y[n]$ to the nearest non-negative integer" mechanism for the sake of simplicity.
  - Such a mechanism may destroy the self-similar structure of the output sequence, if the mean traffic rate of the filter input is too small.

Truncation Analysis and Integerization

- We propose to round the accumulative function of $y[n]$ instead of $y[n]$ itself.
- Let

$$z[n] = \left[F_Y[n] - F_Y[n-1]\right]^+,$$

where $[x]^+ = \max\{x, 0\}$, and

$$F_Y[n] = \left\lfloor \sum_{k=1}^{n} y[k] \right\rfloor .$$

- Notably, the operator $[\cdot]^+$ can be removed if $y[1], y[2], y[3], \ldots$ are non-negative.
- We will examine the effect of the new mechanism by simulations.

Truncation Analysis

Define the truncated forward filter with truncation window W as:

$$h_f[n;W] = \begin{cases} \dfrac{\Gamma(n+H-0.5)}{\Gamma(n+1)\Gamma(H-0.5)}, & \text{for } 0 \le n < W; \\ \\ 0, & \text{otherwise} \end{cases}$$

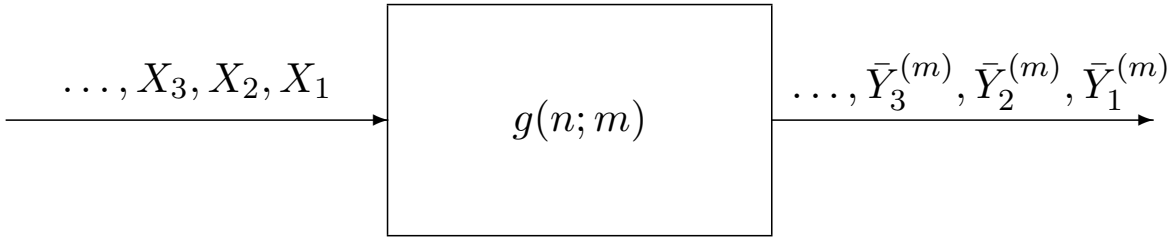
Define the truncated reverse filter with truncation window W as:

$$h_r[n;W] = \begin{cases} \dfrac{(H-0.5)\cdot\Gamma(n-H+0.5)}{\Gamma(n+1)\Gamma(1.5-H)}, & \text{for } 1 \le n < W; \\ \\ 0, & \text{otherwise.} \end{cases}$$

Variance-equivalent m-averaged process

- A *variance-equivalent m-averaged process* $\bar{Y}_1^{(m)}, \bar{Y}_2^{(m)}, \bar{Y}_3^{(m)}, \ldots$ of a random process $X_1, X_2, X_3, \ldots$ is an output process through the filter $g[n; m]$.
- $g[n; m]$ defined as:

$$g[n;m] = \begin{cases} \dfrac{1}{m}, & \text{for } 0 \le n < m; \\ \\ 0, & \text{otherwise.} \end{cases}$$



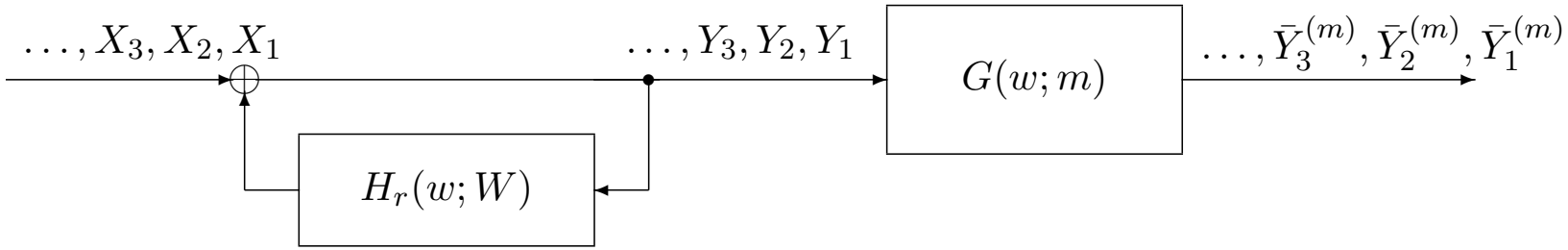
Variance-equivalent m-averaged process

- The impact of the truncation windows size W between the forward filter and the reverse filter can be characterized through the derivation of the marginal variance $B_m(0;W)$ of the m-averaged filter output process due to an i.i.d Poisson input.

Truncation Analysis for the Reverse Filter

Let $L(w) = \dfrac{G(w;m)}{1 - H_r(w;W)}$.

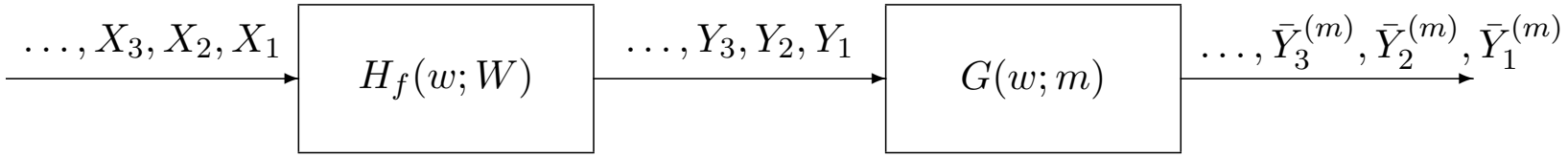
i.i.d. process



$$
\begin{aligned}
B_m^{(r)}(0;W) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} |L(w)|^2 dw \\
&= \frac{1}{2\pi}\int_{-\pi}^{\pi} \frac{|G(w;m)|^2}{|1 - H_r(w;W)|^2} dw \\
&= \frac{1}{2\pi m^2}\int_{-\pi}^{\pi} \frac{1}{|1 - H_r(w;W)|^2}\frac{\sin^2(mw/2)}{\sin^2(w/2)} dw.
\end{aligned}
$$

Truncation Analysis for the Forward Filter

i.i.d. process



Let $L_f(w) = H_f(w;W)G(w;m)$.

$$\begin{aligned} B_m^{(f)}(0;W) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} |L_f(w)|^2 dw \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left[\sum_{n=0}^{\infty} \ell_f[n]e^{-jnw}\right]\left[\sum_{n=0}^{\infty} \ell_f[n]e^{jnw}\right] dw \\ &= \sum_{n=0}^{\infty} |\ell_f[n]|^2 \end{aligned}$$

Truncation Analysis for the Forward Filter

A) $m \leq W$**.**

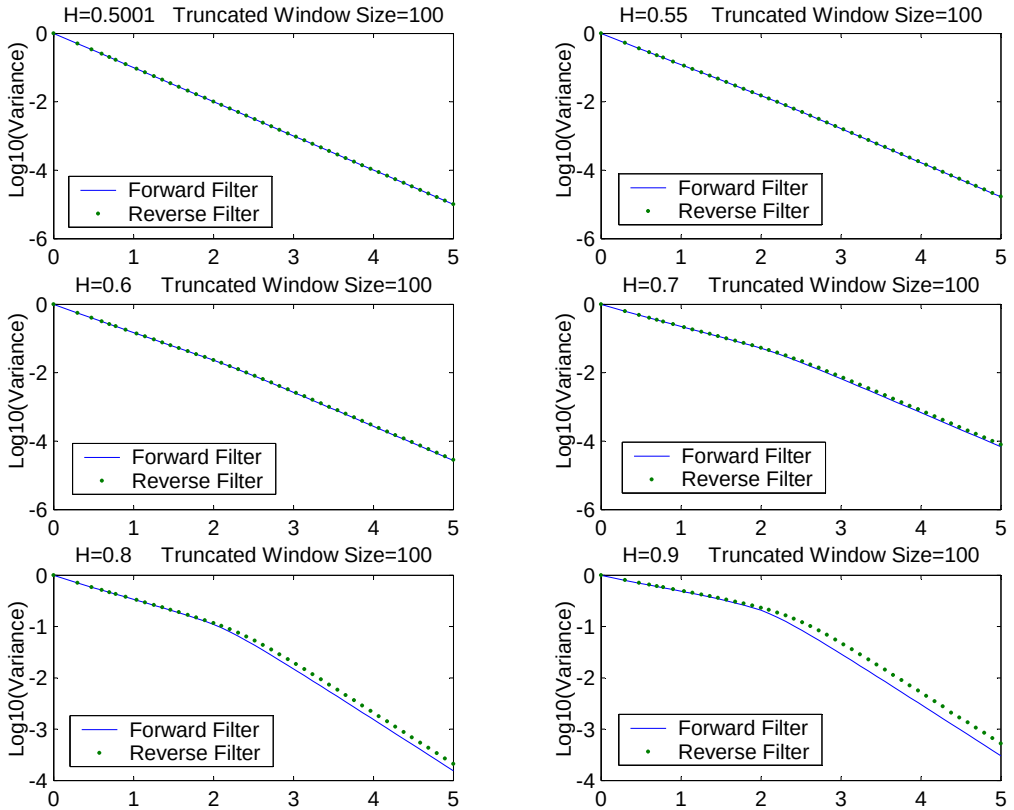
$$B_m^{(f)}(0;W) = \frac{1}{m^2}\sum_{l=0}^{W-m}\left(\sum_{n=l}^{l+m-1} h_f[n]\right)^2 + \frac{1}{m^2}\sum_{l=0}^{m-2}\left[\left(\sum_{n=0}^{l} h_f[n]\right)^2 + \left(\sum_{n=W-1-l}^{W-1} h_f[n]\right)^2\right]$$

B) $m > W$**.**

$$B_m^{(f)}(0;W) = \frac{1}{m^2}\left[(m-W+1)\left(\sum_{n=0}^{W-1} h_f[n]\right)\right]^2 + \frac{1}{m^2}\sum_{l=0}^{W-2}\left[\left(\sum_{n=0}^{l} h_f[n]\right)^2 + \left(\sum_{n=l+1}^{W-1} h_f[n]\right)^2\right]$$

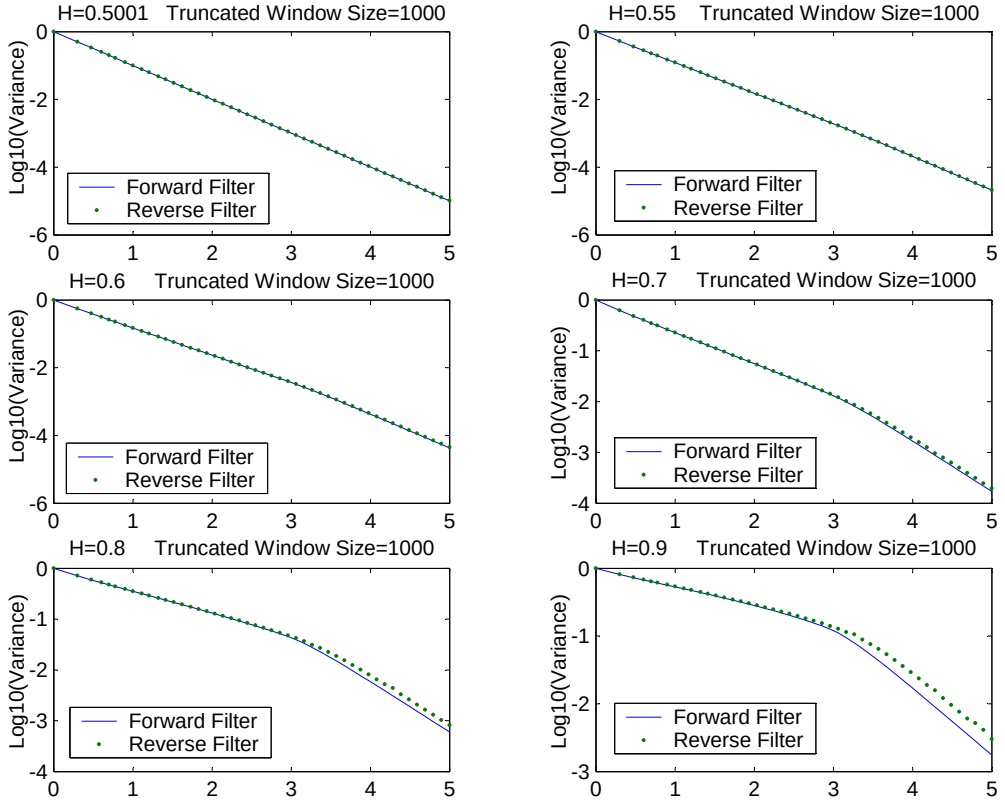
Truncation Analysis

Variance-time analysis ($\log_{10}$ scale) for the truncated reverse- and forward-filter outputs with truncation window $W = 10^2$. The solid (blue) and dotted (green) lines correspond respectively to forward- and reverse-filter results.



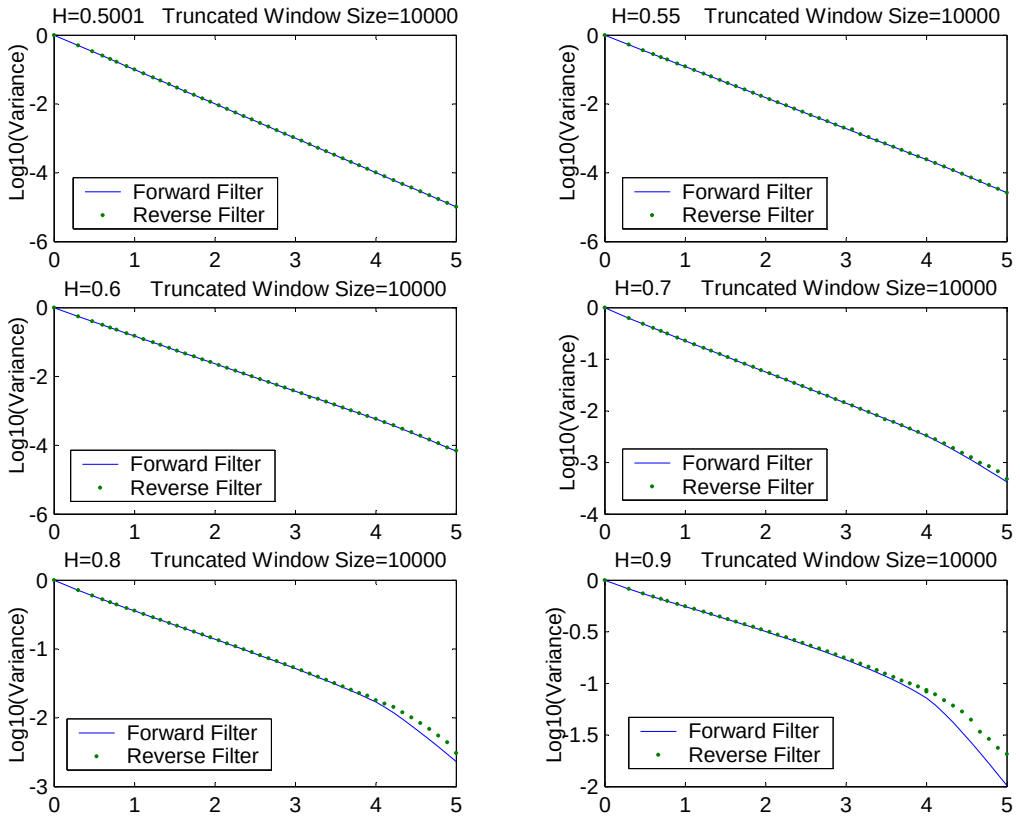
Truncation Analysis

Variance-time analysis ($\log_{10}$ scale) for the truncated reverse- and forward-filter outputs with truncation window $W = 10^3$. The solid (blue) and dotted (green) lines correspond respectively to forward- and reverse-filter results.



Truncation Analysis

Variance-time analysis ($\log_{10}$ scale) for the truncated reverse- and forward-filter outputs with truncation window $W = 10^4$. The solid (blue) and dotted (green) lines correspond respectively to forward- and reverse-filter results.



Truncation Analysis

The self-similar parameters for the synthetic traffics.The best-fit lines are calculated for $W \leq m \leq 10^{0.6}W$. $\hat{H}_f$ and $\hat{H}_r$ represent the obtained values for forward filter approach and reverse filter approach, respectively. Deviations are defined as $D_f = (\hat{H}_f - H)/H$ and $D_r = (\hat{H}_r - H)/H$,where H is the targeted self-similar parameter.

Window Size=10^2				
Targeted H	$\hat{H}_f$	$\hat{H}_r$	D_f	D_r
0.5001	0.5000643	0.5000700	-0.000713	-0.000598
0.55	0.526311	0.528772	-0.043071	-0.038596
0.6	0.548175	0.557430	-0.086375	-0.070949
0.7	0.580904	0.614373	-0.170137	-0.122323
0.8	0.602104	0.671024	-0.247370	-0.161220
0.9	0.615083	0.726966	-0.316574	-0.192259

Truncation Analysis

The self-similar parameters for the synthetic traffics. The best-fit lines are calculated for $W \leq m \leq 10^{0.6}W$. $\hat{H}_f$ and $\hat{H}_r$ represent the obtained values for forward filter approach and reverse filter approach, respectively. Deviations are defined as $D_f = (\hat{H}_f - H)/H$ and $D_r = (\hat{H}_r - H)/H$, where H is the targeted self-similar parameter.

Window Size=10^3				
Targeted H	$\hat{H}_f$	$\hat{H}_r$	D_f	D_r
0.5001	0.5000484	0.5000519	-0.001030	-0.000961
0.55	0.522934	0.525323	-0.049210	-0.044868
0.6	0.541746	0.550603	-0.097090	-0.082328
0.7	0.569467	0.602010	-0.186476	-0.139985
0.8	0.587001	0.656429	-0.266248	-0.179463
0.9	0.597748	0.706871	-0.335836	-0.214587

Truncation Analysis

The self-similar parameters for the synthetic traffics.The best-fit lines are calculated for $W \leq m \leq 10^{0.6}W$. $\hat{H}_f$ and $\hat{H}_r$ represent the obtained values for forward filter approach and reverse filter approach, respectively. Deviations are defined as $D_f = (\hat{H}_f - H)/H$ and $D_r = (\hat{H}_r - H)/H$,where H is the targeted self-similar parameter.

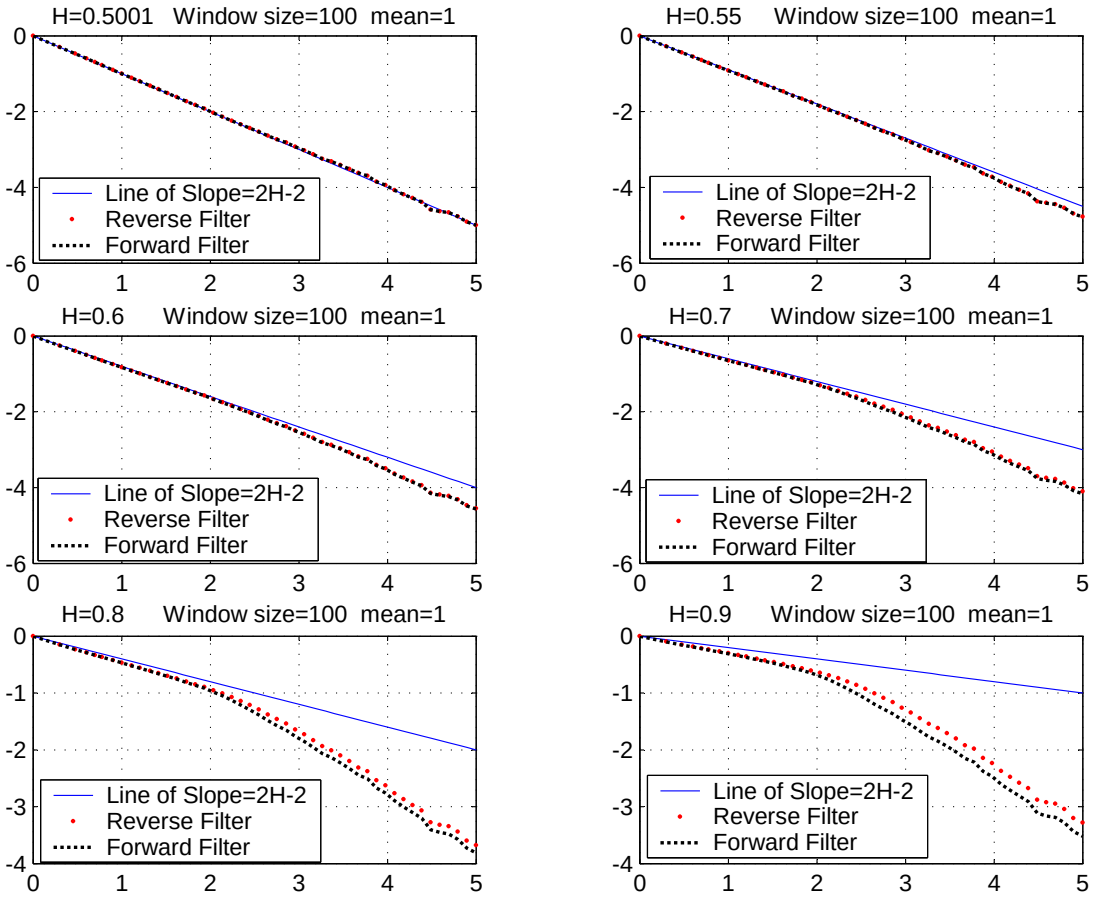
Window Size=10^4				
Targeted H	$\hat{H}_f$	$\hat{H}_r$	D_f	D_r
0.5001	0.5000465	0.5000419	-0.001068	-0.001159
0.55	0.525465	0.527560	-0.044609	-0.040800
0.6	0.546588	0.555105	-0.089020	-0.074825
0.7	0.578034	0.615440	-0.174238	-0.120800
0.8	0.598354	0.700691	-0.252058	-0.124136
0.9	0.610636	0.753034	-0.321515	-0.163295

Simulations of Synthetic Traces

Simulations of Synthetic Traces

- We obtain from numericals that the reverse filter improvement is quite limited to our surprise.
- A larger window size, or a larger input mean rate, or a higher filter parameter H gives more improvement.
- This can be confirmed by taking a close look at the simulation result corresponding to window size $10,000$, input mean rate 100 and filter parameter $H = 0.9$

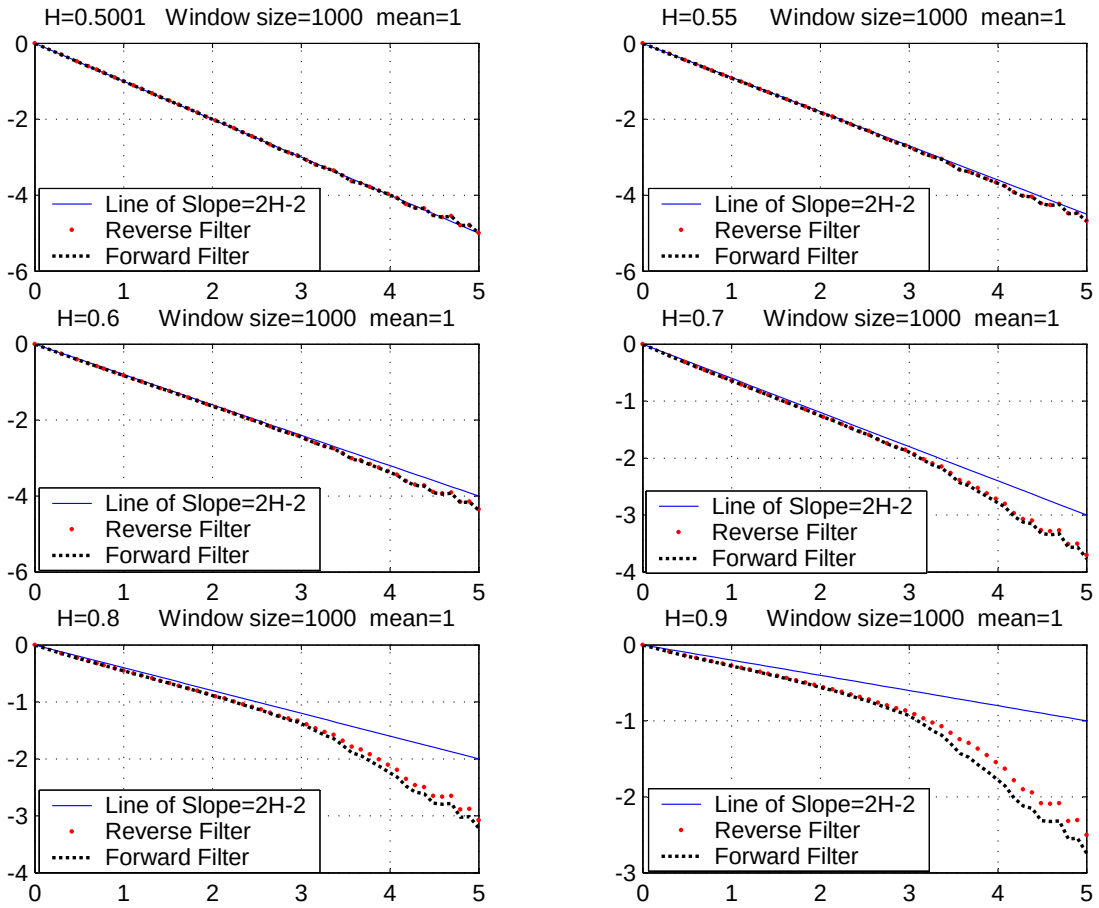
Variance-time plots ($\log_{10}$ scale) for the two filter-based synthetic arrivals with truncation window 10^2 and mean rate 1.



Simulations of Synthetic Traces

Window Size= 10^2 and mean rate= 1				
Targeted H	$\hat{H}_f(m \le W)$	$\hat{H}_r(m \le W)$	$\hat{H}_f(W \le m \le 10^{0.6}W)$	$\hat{H}_r(W \le m \le 10^{0.6}W)$
0.5001	0.501260971	0.501260975	0.515888448	0.516808234
0.55	0.546097230	0.546115922	0.541163694	0.542083290
0.6	0.592068671	0.592204877	0.558707330	0.567850033
0.7	0.682493542	0.683633132	0.586579041	0.620015754
0.8	0.766404866	0.770393771	0.604143124	0.673417427
0.9	0.837857103	0.846922931	0.614666028	0.727600348
Targeted H	$\hat{H}_f(m > W)$	$\hat{H}_r(m > W)$	$\hat{H}_f$(for all m)	$\hat{H}_r$(for all m)
0.5001	0.496599819	0.496601023	0.501348013	0.501349501
0.55	0.500299782	0.500737237	0.523510278	0.524070384
0.6	0.503273069	0.504885843	0.544856931	0.546974512
0.7	0.507426563	0.513397521	0.583449169	0.591667778
0.8	0.509938942	0.522827208	0.615993557	0.634239056
0.9	0.511408850	0.533920803	0.641784779	0.673746092

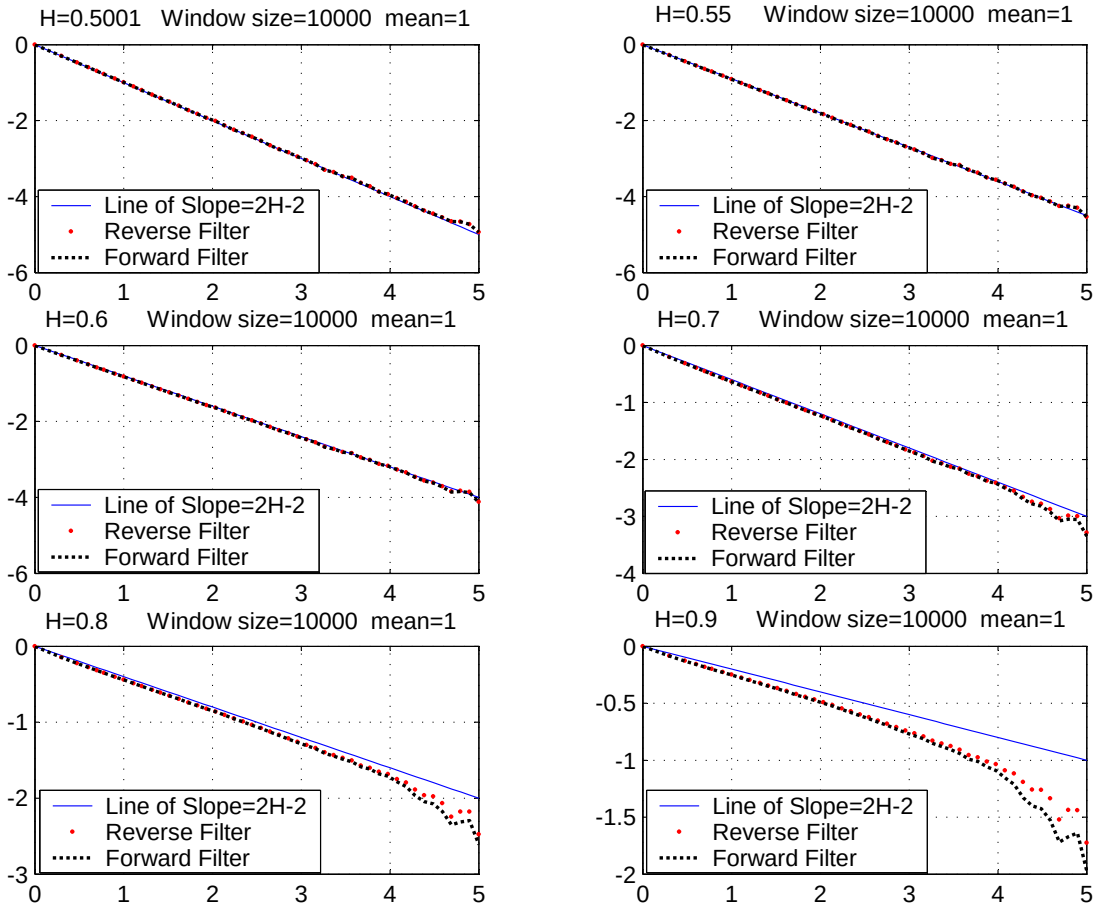
Variance-time plots ($\log_{10}$ scale) for the two filter-based synthetic arrivals with truncation window 10^3 and mean rate 1.



Simulations of Synthetic Traces

Window Size= 10^3 and mean rate= 1				
Targeted H	$\hat{H}_f(m \le W)$	$\hat{H}_r(m \le W)$	$\hat{H}_f(W \le m \le 10^{0.6}W)$	$\hat{H}_r(W \le m \le 10^{0.6}W)$
0.5001	0.498954604	0.498954576	0.478525976	0.478527232
0.55	0.545590738	0.545598865	0.500509826	0.502602652
0.6	0.593268447	0.593332835	0.518321516	0.526495677
0.7	0.687280370	0.687870085	0.543178024	0.574518351
0.8	0.775720702	0.777951755	0.557968425	0.625613037
0.9	0.852199226	0.857655140	0.566363453	0.681486674
Targeted H	$\hat{H}_f(m > W)$	$\hat{H}_r(m > W)$	$\hat{H}_f$(for all m)	$\hat{H}_r$(for all m)
0.5001	0.518747190	0.518746804	0.500920574	0.500920620
0.55	0.525569713	0.526309565	0.536311487	0.536736510
0.6	0.531094180	0.533958460	0.571247273	0.572955153
0.7	0.538934659	0.549870753	0.636851387	0.643810334
0.8	0.543839017	0.567678880	0.695110535	0.711055666
0.9	0.546870991	0.588444618	0.743354416	0.771975380

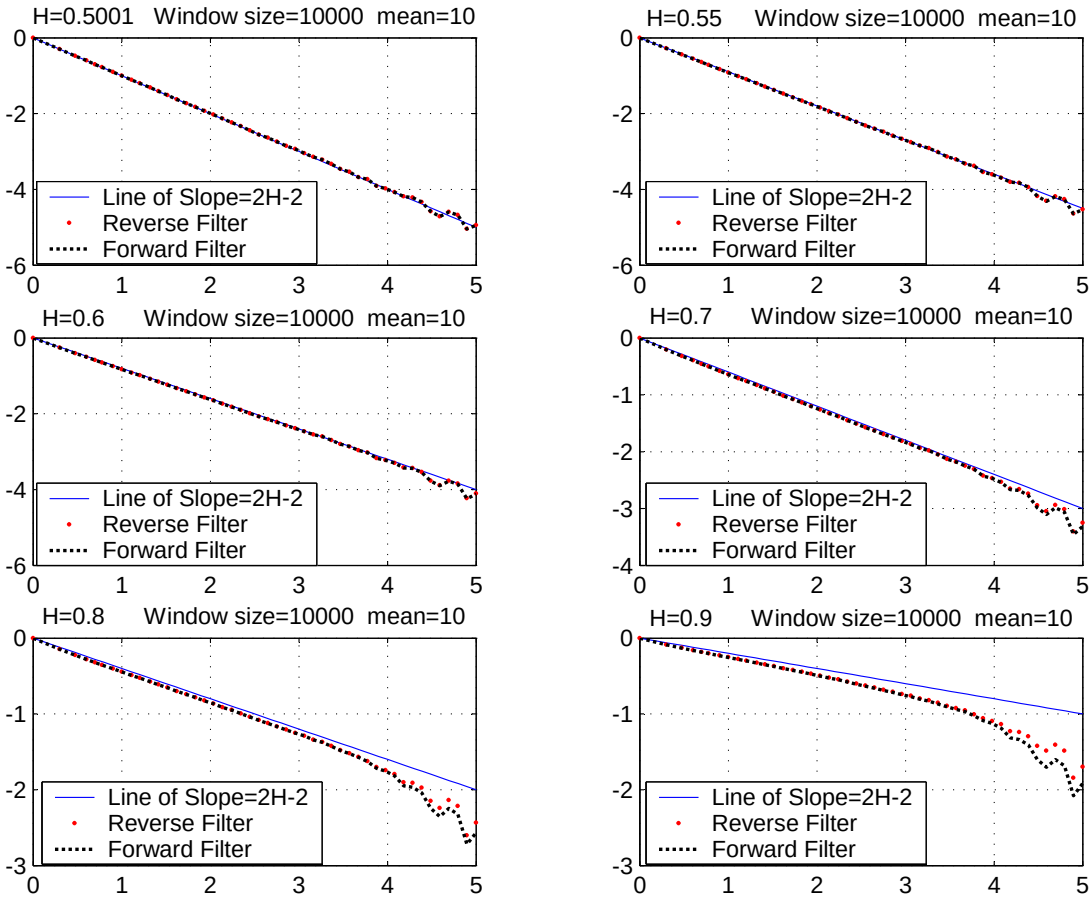
Variance-time plots ($\log_{10}$ scale) for the two filter-based synthetic arrivals with truncation window 10^4 and mean rate 1.



Simulations of Synthetic Traces

Window Size= 10^4 and mean rate= 1				
Targeted H	$\hat{H}_f(m \le W)$	$\hat{H}_r(m \le W)$	$\hat{H}_f(W \le m \le 10^{0.6}W)$	$\hat{H}_r(W \le m \le 10^{0.6}W)$
0.5001	0.502663565	0.502663591	0.499656574	0.499657533
0.55	0.550436362	0.550458536	0.529486644	0.531872041
0.6	0.599215607	0.599329526	0.554802737	0.563995995
0.7	0.695678616	0.696397994	0.593413833	0.627446413
0.8	0.787441046	0.789737858	0.620335402	0.691490788
0.9	0.867678988	0.872544713	0.639188033	0.758955409
Targeted H	$\hat{H}_f(m > W)$	$\hat{H}_r(m > W)$	$\hat{H}_f$(for all m)	$\hat{H}_r$(for all m)
0.5001	0.552278927	0.552279392	0.506475740	0.506475853
0.55	0.566694932	0.568272430	0.551946678	0.552185760
0.6	0.578331075	0.584440386	0.597617025	0.598598318
0.7	0.594684707	0.617913286	0.685729718	0.689869001
0.8	0.604707659	0.655533075	0.766961756	0.776695185
0.9	0.610720742	0.700961849	0.836245417	0.853715090

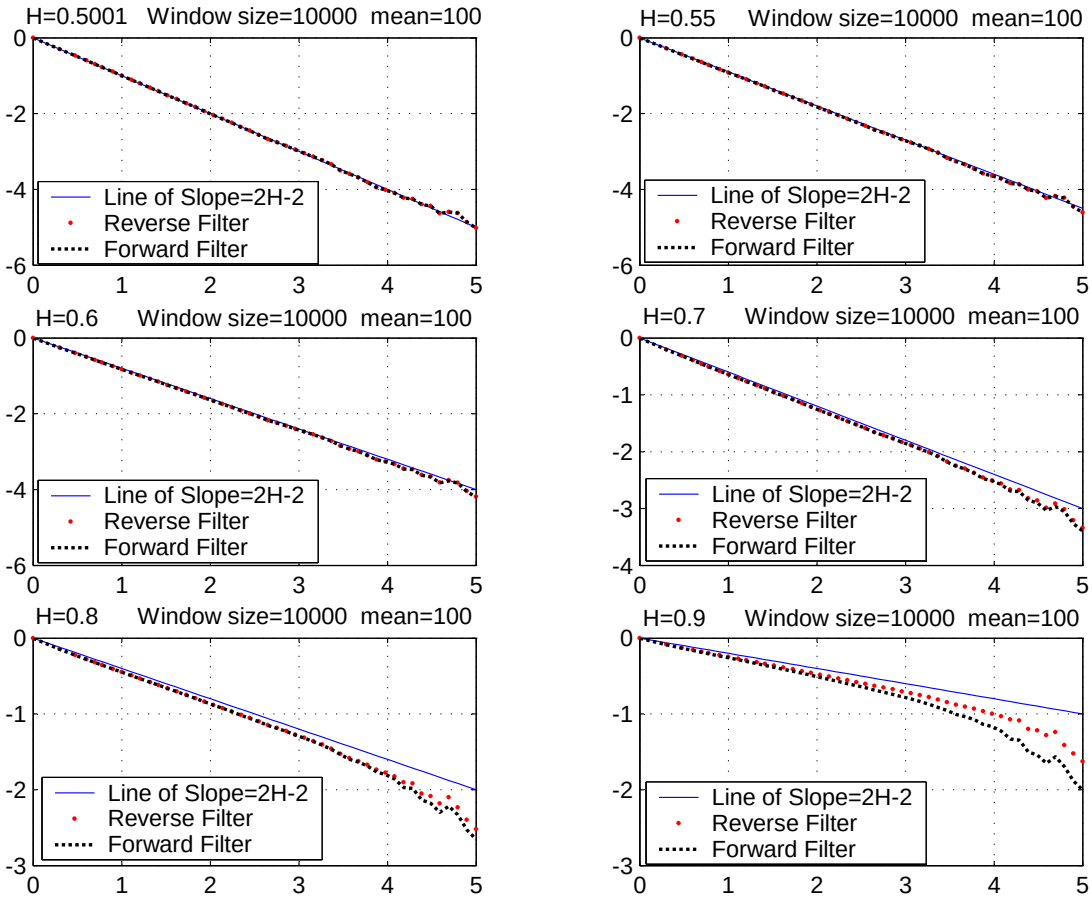
Variance-time plots ($\log_{10}$ scale) for the two filter-based synthetic arrivals with truncation window 10^4 and mean rate 10.



Simulations of Synthetic Traces

Window Size= 10^4 and mean rate= 10				
Targeted H	$\hat{H}_f(m \le W)$	$\hat{H}_r(m \le W)$	$\hat{H}_f(W \le m \le 10^{0.6}W)$	$\hat{H}_r(W \le m \le 10^{0.6}W)$
0.5001	0.502071229	0.502071188	0.369091988	0.369092158
0.55	0.549947402	0.549938034	0.396368318	0.399037077
0.6	0.598857151	0.598836571	0.420805580	0.431325303
0.7	0.695576474	0.695684350	0.460111180	0.500602906
0.8	0.787477330	0.788168744	0.488704018	0.577046316
0.9	0.867614568	0.870113146	0.509177667	0.667264948
Targeted H	$\hat{H}_f(m > W)$	$\hat{H}_r(m > W)$	$\hat{H}_f$(for all m)	$\hat{H}_r$(for all m)
0.5001	0.493786355	0.493785923	0.500678600	0.500678580
0.55	0.511942270	0.513691703	0.545710003	0.545921730
0.6	0.527349180	0.534175325	0.591022843	0.591895027
0.7	0.527349180	0.576389347	0.678637982	0.682326966
0.8	0.565683305	0.621857514	0.759549886	0.768129352
0.9	0.575360873	0.674978596	0.828557137	0.844703731

Variance-time plots ($\log_{10}$ scale) for the two filter-based synthetic arrivals with truncation window 10^4 and mean rate 100.



Simulations of Synthetic Traces

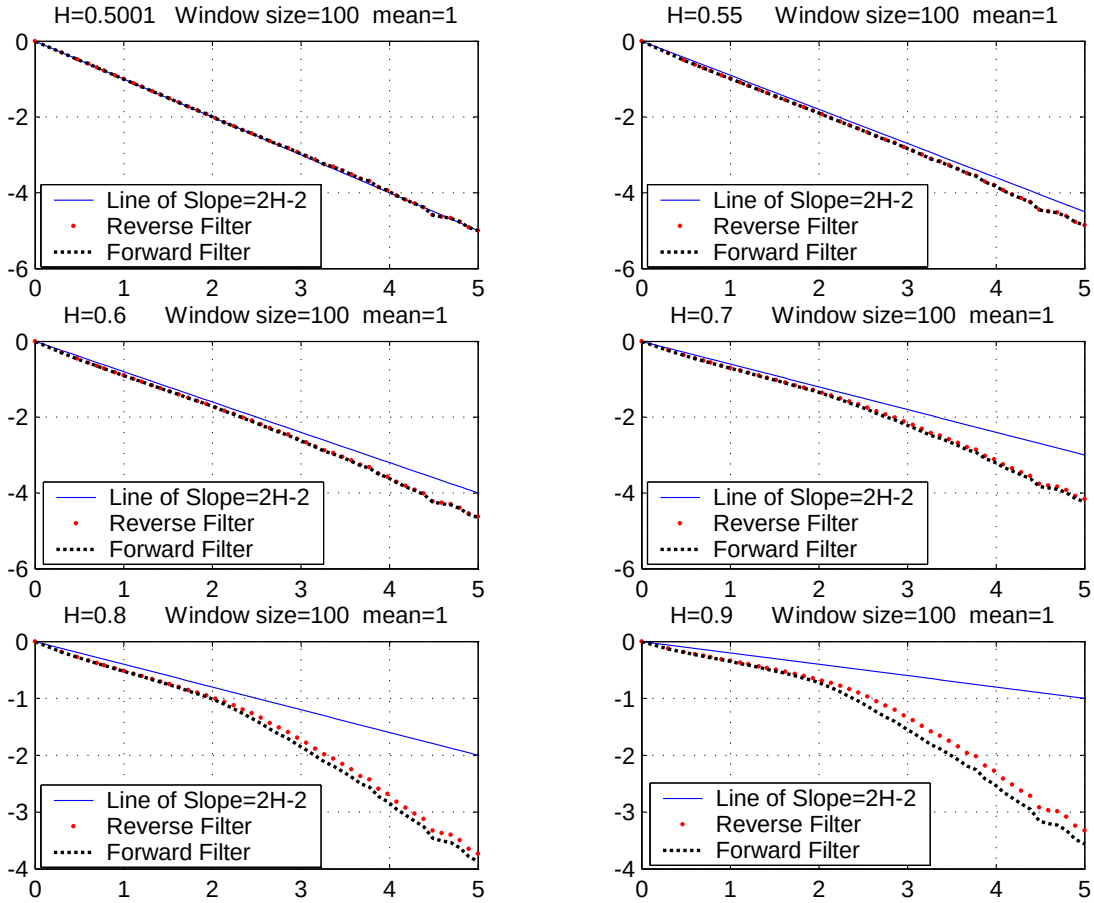
Window Size= 10^4 and mean rate= 100				
Targeted H	$\hat{H}_f(m \le W)$	$\hat{H}_r(m \le W)$	$\hat{H}_f(W \le m \le 10^{0.6}W)$	$\hat{H}_r(W \le m \le 10^{0.6}W)$
0.5001	0.497246198	0.497246120	0.511465019	0.511466758
0.55	0.544548086	0.544563855	0.531073160	0.532938024
0.6	0.592923733	0.593015851	0.547128319	0.554365159
0.7	0.688771672	0.689408190	0.570131197	0.598184174
0.8	0.780205286	0.782078295	0.584353974	0.653841934
0.9	0.860654818	0.878830584	0.592574436	0.748523990
Targeted H	$\hat{H}_f(m > W)$	$\hat{H}_r(m > W)$	$\hat{H}_f$(for all m)	$\hat{H}_r$(for all m)
0.5001	0.539545377	0.539543479	0.501524117	0.501524117
0.55	0.557843286	0.559306594	0.546162490	0.546380935
0.6	0.572982555	0.578642629	0.591086999	0.591998771
0.7	0.594811150	0.616269697	0.678042970	0.681997106
0.8	0.608365390	0.657394621	0.758625797	0.768168948
0.9	0.616268404	0.693442538	0.827941415	0.861618275

Simulations of Synthetic Traces

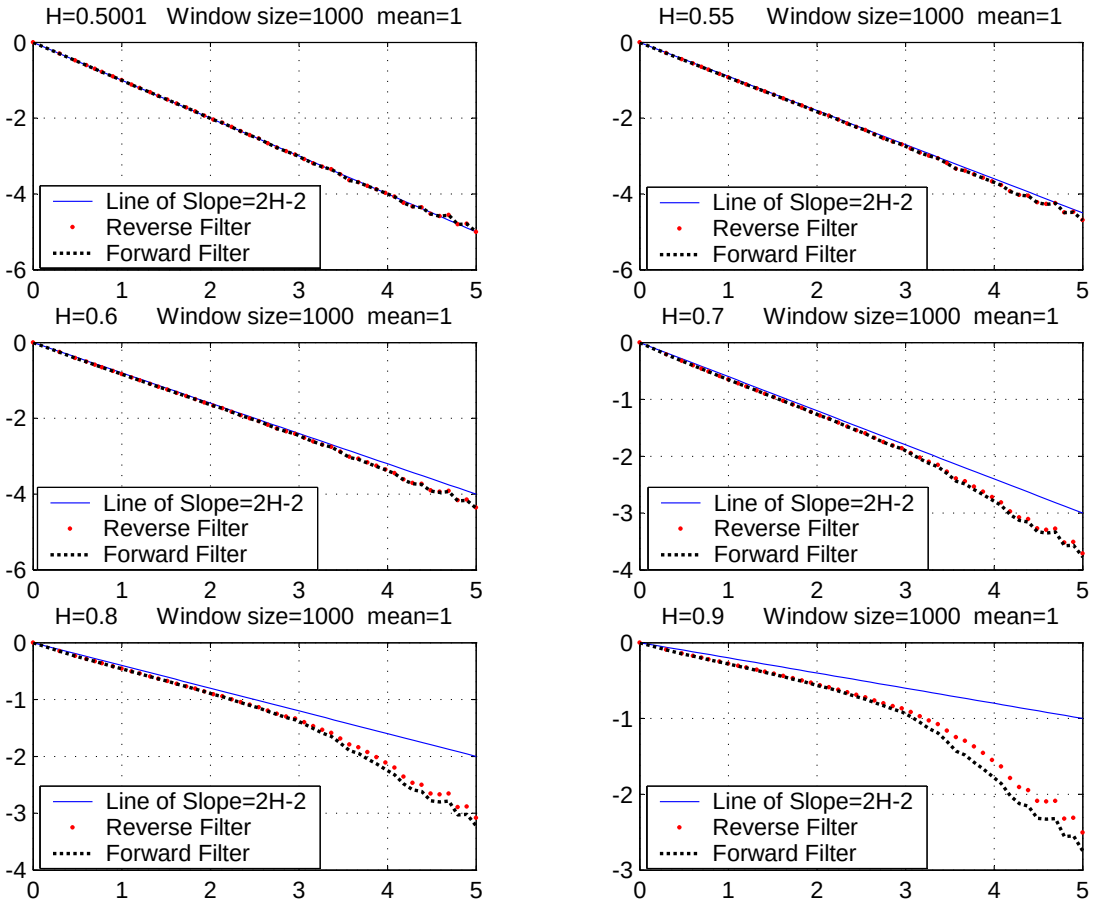
Integerization

- Our "integerization" method will not destroy the self-similar structure of the output sequence.
- The self-similar structure of the output sequence is almost identical to the original no intergerization sequence.

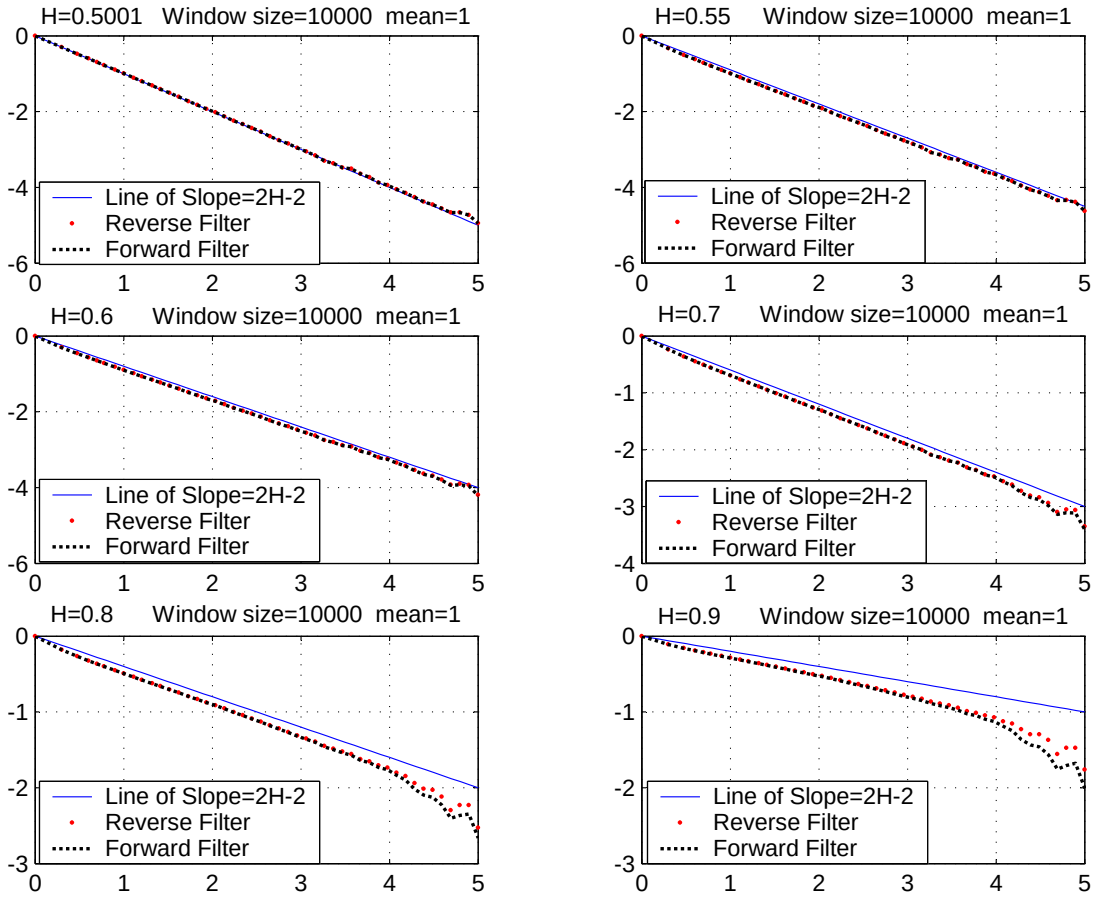
Variance-time plots ($\log_{10}$ scale) for the two filter-based **integerized** synthetic arrivals with truncation window 10^2 and mean rate 1.



Variance-time plots ($\log_{10}$ scale) for the two filter-based **intergerized** synthetic arrivals with truncation window 10^3 and mean rate 1.



Variance-time plots ($\log_{10}$ scale) for the two filter-based **intergerized** synthetic arrivals with truncation window 10^4 and mean rate 1.



Conclusion

- From our simulations, we obtain that the reverse filter traffic synthesizer can give a more self-similar traffic than the one based on forward filter approach, especially at larger H close to 1.
- The model is **parsimonious** in the number of model **parameters**. Specifically, it only depends on three parameters:
  - H : control the bustiness and autocorrelation of the synthesized traffic
  - λ : define the mean of the synthesized traffic
  - W : determines not only the length of the filter but also the valid aggregation size of preserved self-similar nature.
- Compared to the Paxson IFFT, and Ledesma and Liu FFT, our model provides additional advantages that the synthetic traffic can be generated **on the fly**, and is always **non-negative**.

Conclusion

- **Complexity**
  - The known fastest algorithm, Paxson IFFT, requires $(n/2)(\log_2(n)+2)$ complex multiplications, where n is the length of the generated trace.
  - Hua's forward filter requires $n \times W$ complex multiplications, where W represents the truncation window size.
  - Our reverse filter requires the same complex multiplications as Hua's forward filter.
- After analytically analyzing our approach based on variance-time test, we conclude that our method guarantees the generation of a traffic with desired degree of self-similarity beyond the intended range.