J Geod (2017) 91:985–994
DOI 10.1007/s00190-017-1002-5

ORIGINAL ARTICLE

Improved efficiency of maximum likelihood analysis of time series with temporally correlated errors

John Langbein[1]

Received: 20 January 2016 / Accepted: 31 January 2017 / Published online: 11 February 2017

Abstract Most time series of geophysical phenomena have temporally correlated errors. From these measurements, various parameters are estimated. For instance, from geodetic measurements of positions, the rates and changes in rates are often estimated and are used to model tectonic processes. Along with the estimates of the size of the parameters, the error in these parameters needs to be assessed. If temporal correlations are not taken into account, or each observation is assumed to be independent, it is likely that any estimate of the error of these parameters will be too low and the estimated value of the parameter will be biased. Inclusion of better estimates of uncertainties is limited by several factors, including selection of the correct model for the background noise and the computational requirements to estimate the parameters of the selected noise model for cases where there are numerous observations. Here, I address the second problem of computational efficiency using maximum likelihood estimates (MLE). Most geophysical time series have background noise processes that can be represented as a combination of white and power-law noise, $1/f^{\alpha}$ with frequency, f. With missing data, standard spectral techniques involving FFTs are not appropriate. Instead, time domain techniques involving construction and inversion of large data covariance matrices are employed. Bos et al. (J Geod, 2013. doi:10.1007/s00190-012-0605-0) demonstrate one technique that substantially increases the efficiency of the MLE methods, yet is only an approximate solution for power-law indices >1.0 since they require the data covariance matrix to be Toeplitz. That restriction can be removed by simply forming a data filter that adds noise processes rather than combining them in quadrature. Consequently, the inversion of the data covariance matrix is simplified yet provides robust results for a wider range of power-law indices.

Keywords GNSS · Error · Temporal correlation · Time series

1 Introduction

It has been recognized that many time series of geophysical phenomena include background noise processes that exhibit temporal correlations (Agnew 1992). These correlations can be characterized by computing a power spectrum and recognizing that at the highest frequencies, the power is frequency independent and at the lower frequencies, the power can be represented by power-law noise, $1/f^{\alpha}$. From these time series, various coefficients are estimated along with their standard errors. For example, positions determined from global navigation satellite system (GNSS) measurements can be used to estimate site velocities, offsets due to earthquakes, and/or rate changes due to transient deformation from volcanic sources. Yet, implementing any noise model that represents anything more complex than white noise, or, normally distributed, Gaussian error, becomes a computationally inefficient problem in least squares regression since it involves inverting a large, square data–covariance matrix with the number of operations that scale with n^3, where n is the number of observations. In many applications, convenience dictates that the covariance matrix becomes diagonal and standard, weighted least squares is used to estimate the parameters of interest, for example, the velocity. Then, to

✉ John Langbein
langbein@usgs.gov

[1] Earthquake Science Center, US Geological Survey, Menlo Park, CA, USA

estimate the standard error in velocity, empirical relationships, such as those found in Mao et al. (1999), are used to quantify the error. However, this method could provide erroneous estimates of both the velocity and its standard error as it makes several assumptions. If the velocity is computed on the basis of uncorrelated data, yet the background noise has strong temporal correlations approaching that of a random walk, where $\alpha = 2$, the estimated velocity will be biased. And, if the background noise spectrum does not conform to the "rule of thumb" applied to an empirically derived relation, the standard error in velocity could also be incorrect.

Over the past two decades, a number of papers have outlined methods to better estimate the functional parameters that describe a time series and simultaneously estimate the components of an assumed model of the background noise. Most work has revolved around using maximum likelihood estimators (MLE) to optimize both the fit to the data of the function that describes the time dependence and the noise model that describes the data covariance matrix. The initial work by both Williams et al. (2004) and Langbein and Johnson (1997) produced similar algorithms with the only significant difference being the types of functions that could represent the time dependence of the observations. Later, Bos et al. (2008, 2013) made improvements on computational efficiency. Bos et al. (2008) transform the observations into first differences and note the Toeplitz nature of the data covariance and implement a fast inversion method to obtain the inverse covariance. However, this requires that the time series have no gaps, which is usually not the case with field measurements. On the other hand, Bos et al. (2013) present another method that allows for gaps in the time series but requires the data covariance matrix to approximate a Toeplitz matrix; this restricts the power-law index, α, to be ≤ 1 but in practice, allows $\alpha < 1.6$, which excludes random walk. Recent work by Bos and Fernandes (2015) extends the Toeplitz approximation to encompass random-walk noise by using the generalized Gauss–Markov noise model (Langbein 2004) and selecting the Gauss–Markov period longer than the length of the time series.

In parallel, there are two other methods for quantifying temporal correlation. Amiri-Simkooei et al. (2007) uses least squares, variance component estimation to determine the components of a noise model. And, Hackl et al. (2011) employed Allan variance of rate to determine the appropriate noise model but noted that MLE provides more robust results.

Langbein (2004) provides more discussion on the various forms and implementations of power-law noise, while Williams (2003) examines the impact of colored or power-law noise on the estimated uncertainties of rates. For more discussion of fractional, power-law noise, the reader is directed to Hosking (1981) and Kasdin (1995). Finally, Langbein (2012) provides some guidance about assessing models of colored noise obtained from MLE analysis. In particular, it is difficult to confidently extract the random-walk contribution due to both the possible presence of flicker noise and limited length of the time series. Dmitrieva et al. (2015) present an alternative method, namely a network approach, to extract the random-walk contribution of noise to the data.

Although the power-law noise models used to quantify the temporal correlations are important, the research presented here takes off from Bos et al. (2013) and removes the restriction that the covariance matrix representing power-law noise conforms to a Toeplitz matrix. The main result of Bos et al. (2013) is the decomposition of the data covariance into two parts, one representing the data covariance for a time series with no gaps and a second representing the correction to the data covariance for the missing observations.

Instead, I use the decomposition presented by Bos et al. (2013) and use a different assumption to construct the underlying noise model, which yields similar computational efficiency on par with Bos et al. (2013). Previously, if there were two or more sources for modeled noise, these noise sources were taken to be independent and added in quadrature. However, I present a method that assumes that a single, white noise source is filtered such that it represents colored noise. The filter is constructed by adding various constituents that comprise colored noise and then uses a simple algorithm to construct the inverse filter using deconvolution to create the inverse of the data covariance. Comparison with code from Langbein (2004) indicates up to a factor of 50 increase in speed for large, 4000 observation data sets with few data gaps. This results in a code that runs nearly as fast as that of Bos et al. (2013).

The following quickly reviews least squares and the role of the data covariance. The noise model that I propose is introduced along with the method of Bos et al. (2013) for working with missing observations. Then, using simulated data, I make comparisons of various coefficients estimated using the traditional noise model and the one I propose here.

2 Revised data covariance model

Prior to discussing the revised data covariance, I will quickly review least squares where a design matrix, A, is constructed that relates the observations, d, to the parameters, x, which are estimated by scaling the design matrix to fit the data; $\breve{d} = \breve{A}x + \breve{e}$, where $\breve{d}$ and $\breve{e}$ represent the data and their error that contain $(n - m)$ observations, where m is the number of missing observations or gaps. The size of $\breve{A}$ is $(n - m)$ by p, where p is the number of unknown parameters in x. To estimate the value of x using least squares, one calculates:

$$\hat{x} = (\breve{A}^{\mathrm{t}}\breve{C}^{-1}\breve{A})^{-1}\breve{A}^{\mathrm{t}}\breve{C}^{-1}\breve{d} \tag{1}$$

where $\breve{C}$ is the data covariance matrix. The data residuals, r are calculated by:

$$\breve{r} = \breve{d} - \breve{A}\hat{x} \tag{2}$$

Finally, the logarithm of the Gaussian probability function is:

$$\ln(\rho(r, C)) = -0.5\Big[(n-m)\ln(2\pi) + \ln(\det(\breve{C})) + \breve{r}^{\mathrm{t}}\breve{C}^{-1}\breve{r}\Big] \tag{3}$$

The process of maximizing the probability or likelihood is iterative, which is initialized by assuming a model for the data covariance, estimating the model parameters, x, using Eq. (1), computing the misfit to the model, Eq. (2), and evaluating the likelihood, Eq. (3). This sequence is known as maximum likelihood estimation. Equation (3) measures both the size of the data covariance, $\det(\breve{C})$, and the normalized misfit, $\breve{r}^{\mathrm{t}}\breve{C}^{-1}\breve{r}$. Using a simplex algorithm of Nelder and Mead (1965), adjustments are made to the parameters that are used to compute the data covariance until Eq. (3) achieves a maximum. Note that the model parameters, x, and the residuals, $\breve{r}$, are updated for each iteration to minimize the potential for bias.

In previous work with constructing the data covariance, the data noise is constructed as a convolution between a filter, f_i and white noise, w_i;

$${}_k e_i = \sum_{j=i}^{n} {}_k f_{i-j}\ {}_k w_j \tag{4}$$

where index k represents separate error sources. For white noise, the filter is $a\delta(i)$ and for random walk, the filter is a Heaviside function, $f_i = bh(t)$. Other filter functions including flicker, power-law, and Gauss–Markov processes can be found in Langbein (2004) and Hosking (1981). To construct the data covariance, each error source is assumed to be independent and they are summed in quadrature,

$$e^2 = {}_1e^2 + {}_2e^2 + \cdots + {}_k e^2 \tag{5}$$

where ${}_1e$ might represent the contribution from white noise and the remaining ${}_ke$ are contributions from temporally correlated noise. For instance, the data covariance matrix for a combination of white and random-walk noise with amplitudes a and b, respectively, will look like:

$$\begin{vmatrix} a^2+b^2 & b^2 & b^2 & b^2 \\ b^2 & a^2+2b^2 & 2b^2 & 2b^2 \\ b^2 & 2b^2 & a^2+3b^2 & 3b^2 \\ b^2 & 2b^2 & 3b^2 & a^2+4b^2 \end{vmatrix} \tag{6}$$

To take advantage of partitioning of the covariance matrix between a time series with no gaps and the gappy parts proposed by Bos et al. (2013), I propose an alternative method, which I term additive noise, is to construct the data error by forming the sum

$$e_i = \sum_{j=i}^{n} \sum_{k=1}^{K_{\max}} {}_k f_{i-j}\ w_j \tag{7}$$

where the final filter is a sum of a series of different filters and convolved with a single source of white noise; consequently, this composition inserts crosscorrelation between the constituent filters. For a noise model of white noise and random-walk noise added, as prescribed by Eq. (7), the covariance matrix becomes:

$$\begin{vmatrix} a^2+b^2+2ab & b^2+ab & b^2+ab & b^2+ab \\ b^2+ab & a^2+2b^2+2ab & 2b^2+ab & 2b^2+ab \\ b^2+ab & 2b^2+ab & a^2+3b^2+2ab & 3b^2+ab \\ b^2+ab & 2b^2+ab & 3b^2+ab & a^2+4b^2+2ab \end{vmatrix} \tag{8}$$

where ab is the crosscorrelation between the two filters. Traditionally, both of these two matrices are inverted using Cholesky decomposition and this works well with missing observations. However, in the case of the second, where the filter function is a summation of several functions, the inverse of the data covariance is constructed by first, recognizing that for continuous functions

$$\delta(t) = f(t) * f^{-1}(t) \tag{9}$$

or the convolution of the filter function with its inverse is the delta function, or in terms of discrete samples, the inverse filter is

$$f_i^{-1} = \begin{cases} 1/f_i & \text{if } i = 1; \\ -1.0\left[\sum_{j=i-1}^{1} f_j^{-1} f_{i+1-j}\right]/f_1 & \text{if } i > 1 \end{cases} \tag{10}$$

But, this operation only works with data having no gaps. With the assumption of no gaps, the data covariance matrix is constructed, $C = FF^{\mathrm{t}}$ with

$$F = \begin{vmatrix} f_1 & 0 & 0 & 0 & .. \\ f_2 & f_1 & 0 & 0 & .. \\ f_3 & f_2 & f_1 & 0 & .. \\ f_4 & f_3 & f_2 & f_1 & .. \\ .. & .. & .. & .. & .. \end{vmatrix} \tag{11}$$

Likewise, the inverse of the data covariance is $C^{-1} = F^{-1^{\mathrm{t}}}F^{-1}$.

So far, the above construction of the covariance matrix assumes that there are no missing data. Yet, typical time series will have missing observations. Previous work by Langbein and Johnson (1997), and Williams et al. (2004) have treated the case of missing data by deleting the rows of F that correspond to the missing observations. The resulting covariance matrix, $\breve{C}$, is a $(n-m)$ *by* $(n-m)$ matrix.

However, for the case with gappy data, Bos et al. (2013) present a method that partitions the inverse covariance matrix

into two. The first part is the inverse of the covariance of the data with no gaps, which is rapidly computed using Eq. (10). The second part provides a correction to the inverse of the data covariance due to gaps in the data. The correction is:

$$\breve{r}^{\mathrm{t}}\breve{C}^{-1}\breve{r} = {r_{\mathrm{o}}}^{\mathrm{t}}[C^{-1} - C^{-1}M(M^{\mathrm{t}}C^{-1}M)^{-1}M^{\mathrm{t}}C^{-1}]r_{\mathrm{o}} \quad (12)$$

and

$$\ln(\det(\breve{C})) = \ln(\det(C)) + \ln(\det(M^{\mathrm{t}}C^{-1}M)) \quad (13)$$

where the n by m matrix M selects the columns in C^{-1} for which there are missing data. Each column of M consists of $n-1$ zeros and a value of 1 at the row corresponding to a missing datum in C^{-1}. Consequently, $M^{\mathrm{t}}C^{-1}M$ represents elements corresponding to missing data. Likewise, r_{o} is a vector of length n having undefined values at elements representing missing data and otherwise being the difference between the observed data and their predicted values. The second, lengthy term of Eq. (12) has the property of setting to zero the rows and columns of C^{-1} corresponding to the missing data in r_{o}, and making a correction to C^{-1} due to the missing observations. Consequently, the "missing data" included in r_{o} are nullified when Eq. (12) is evaluated. The proof of this property is provided in the Appendix of Bos et al. (2013).

With no missing observations, Eq. (12) can be rapidly evaluated since C^{-1} is computed from Eq. (10). With a few missing observations, the size of the second term in Eq. (12) is small and the inverse of $(M^{\mathrm{t}}C^{-1}M)$ is computed using Cholesky decomposition along with the rapid computation of C^{-1}. However, at some point, the number of missing observations becomes large enough such that the time required to evaluate the second term of Eq. (12) dominates. In addition, using the simple sums for representing data error, the $\ln(\det(C))$ is simply $n\ln(f_1)$.

The actual implementation of estimating the model parameters and the misfits of the model prediction to the data through Eqs. (1) and (12) is found in the "Appendix." I found it more efficient to regroup the numerous matrices representing the observation equation, covariance, and missing-data operator by exploiting the ability of simple sums comprising the noise model filter to be convolved with the data and model equations.

3 Comparison of two covariance models and inversion algorithms

In this study, the computer program developed by Langbein (2004) was revised in two ways, first to implement the different model of data covariance based upon simple sums rather than quadrature addition, and second to implement the Bos et al. (2013) algorithm, Eqs. (12) and (13), that invert the data covariance with missing observations. This new version, *est_noise7.22*, not only allows a choice between the faster option prescribed by Bos et al. (2013) and the standard method of inverting the data covariance with Cholesky decomposition, it can also optimize the data covariance based on the quadrature addition of noise (Langbein 2004) which is called *est_noise6.50*. The three modes, *7.22f*, *7.22c*, and *7.22n*, allowed by *est_noise7.22* are compared (Table 1). Further comparison is made with software that implements Bos et al. (2013) found at http://segal.ubi.pt/hector/, which also has two options: one using inversion of the data covariance with Cholesky decomposition, *HecC*, and the second implementing both the covariance adjustment for missing data and the fast inverter using the properties of the Toeplitz matrix, *Hec*.

For comparison, I created several sets of simulated data each having a noise model consisting of a combination of white noise and power-law noise added in quadrature (Eq. 5) although additive noise would work OK, too. I specified the white noise of 0.7 mm and a power-law noise with an index and amplitude of 1.5 and 3.0 mm/year$^{1.5/4}$ [Langbein 2004, Eqs. (9) and (10)]. In addition, each time series have a rate of zero and no prescribed offsets. Each simulated set starts with 4000 points of daily samples (10.95 years) with no gaps. For each simulated set, subsets were created by randomly removing data simulating gaps randomly spaced within the original 4000 points. The numbers of missing observations were specified to be 0, 5, 10, 20 through 50% of the total 4000 points. For all of these subsets, each program was used to estimate the rate, the amplitude and phase for 2 sinusoids with periods of 365.25 and 182.625 days, and two offsets. Simultaneously, the programs estimated the components of the white and power-law noise and the corresponding standard errors of the time-dependent model.

The results of the comparison are shown in Figs. 1 through 4. To gather preliminary statistics on the spread of the esti-

Table 1 Comparison of three MLE algorithms

Program	est_noise6.50	est_noise7.22			Hector	
ID	6.50	7.22n	7.22c	7.22f	HecC	Hec
Data error	Quad	Quad	Additive	Additive	Quad	Quad
Toeplitz approx.	No	No	No	No	Yes	Yes
Inverse routine	Cholesky	Cholesky	Cholesky	Bos	Cholesky	Bos

Note that the Bos inverse routine is Eq. (12)

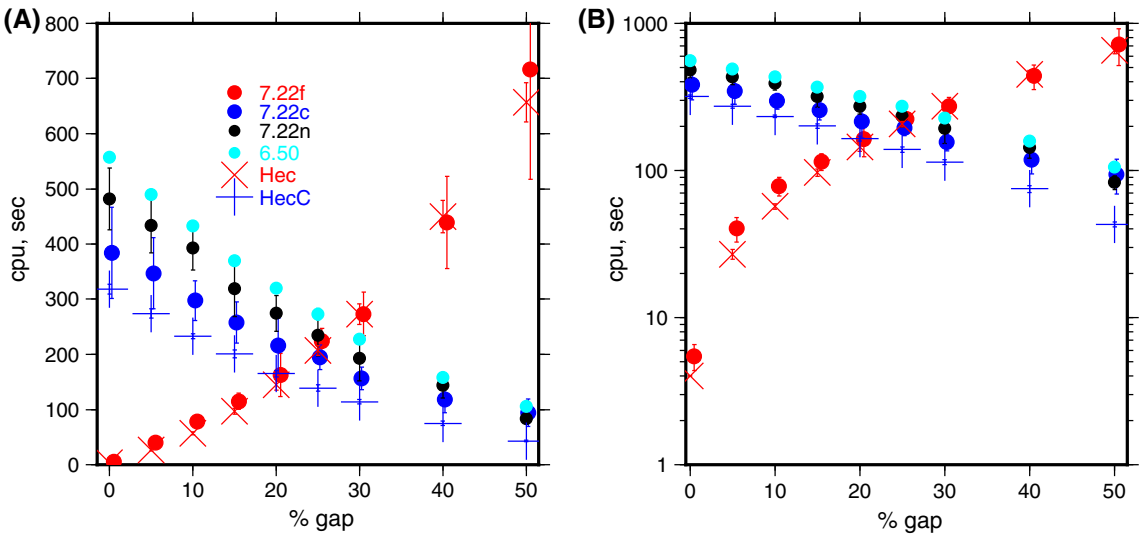

Fig. 1 Statistics of time it takes to estimate both the time-dependent and the noise models from simulations of 10.95 years of daily sampled data. CPU times, or the actual times required for the programs to complete, are plotted as a function of the number of gaps in the data expressed as percentage with 0% having no gaps. The results from six programs or their modes are shown. The vertical bars represent the 25–75% interval of the observed CPU times; if there is *no bar*, then the length of the *bar* is less than the size of the symbol. In (**a**), CPU time is linear but in (**b**), the ordinate is log(CPU time). Note that for the statistics representing 7.22f and 7.22c have been offset slightly in the abscissa for better clarity

mated values, the simulations were run 15 times. These tests were run on a computer with 16 "cores" each running at 2.2 GHz. However, I restricted the operating system to use a single core. The figures show results of a total of six different computations.

The time required for each program to complete is shown in Fig. 1. For a data set with no gaps (0%), the original program, *est_noise6.50* takes just approximately 500 s. In contrast, the revised program, *est_noise7.22* with mode *7.22f*, using both the summed noise (Eq. 7) and inversion of only the left-hand term of Eq. (12) completed in 5 s, or about a factor of 100 speed improvement. However, if the C matrix is inverted using Cholesky decomposition, then mode *7.22c* takes 300 s, or about the same time as the original program, *est_noise6.50*. This is not surprising as these two programs are approximately the same computer code. Between these two codes, there is some difference in the configuration of the Nelder and Mead (1965) algorithm. The computation times for two options of Bos et al. (2013) are also shown. Mode *Hec*, which uses a very fast algorithm to invert Toeplitz matrices, has a 4-s computation time. On the other hand, *HecC*, which uses Cholesky decomposition, has a 320-s computation time.

As the number of gaps increases, the computation times for both mode *7.22f* and *Hec* increase while the computation times for the remaining programs (and modes) decrease. This is because both mode *7.22f* and *Hec* need to evaluate the second term of Eq. (12), while for the other programs or modes, the number of elements of the C matrix decreases. The cpu speed for all five programs becomes about equal between 20 and 25% gaps.

The ability to resolve the power-law index for each algorithm is shown in Fig. 2a. The underlying index of 1.5 is shown as a dashed line. The estimates cluster into three groups. The two versions of *Hector* average to be 1.40. The legacy program *est_noise6.50* and mode *7.22n* of *est_noise7.22*, cluster at 1.45, while modes *7.22f* and *7.22c* average to be 1.53.

Likewise, the estimate of the white noise component from each program is shown in Fig. 2b. For the codes that rely upon quadrature addition of the noise, *Hector*, *est_noise6.50*, and *est_noise7.22n*, they all provide estimates of white noise to within 0.01 mm of the underlying 0.70 mm. On the other hand, the programs that use simple addition for modeling noise yield significantly less apparent white noise. This contrast will be discussed later.

The most relevant comparison for crustal deformation studies is the estimated rate and its standard error using these algorithms, and those results are shown in Fig. 3. For these comparisons, I show the differences in estimated rate and standard error of the five programs and modes relative to the legacy program, *est_noise6.50*. In Fig. 3a, the differences in rates are shown with the estimates from *est_noise7.22* being nearly identical with *est_noise6.50*. In contrast, the estimates from *Hector* are slightly smaller than those from *est_noise6.50*.

The differences in standard error in rate between those computed by *est_noise6.50* and the other programs are shown in Fig. 3b. Mode *7.22n*, as expected, provides the same rate uncertainty as the legacy program that it replaces, *est_noise6.50*. On the other hand, the computed uncertainty from *Hector* and the other two modes of *est_noise7.22* are 0.04 mm/year larger than those estimated by *est_noise6.50* with Hector providing computed uncertainties closer to *est_noise6.50*. This represents approximately a 15% difference from the expected rate uncertainty of 0.27 mm/year.

Along with estimating rate, the ability to estimate offsets and their standard errors is also an important factor to examine; this is shown in Fig. 4. Like the rate comparison in Fig. 3, the offset statistics are the differences relative to *est_noise6.50*. Given that the simulated data had 0.01 mm resolution, the differences shown for offsets are at the same resolution as the simulated data and are 2% of the expected offset uncertainty of 0.50 mm; the offset estimates are independent of the underlying method for construction of the data covariance.

Both Figs. 3 and 4 show the differences in estimated rates and offsets relative to the legacy program. Not shown are the actual values of rate, offset, and standard errors. For the simulations, all had prescribed zero for the rate and offset. Yet, all of the MLE codes estimated a nonzero rate and offset, but when compared to the standard error of each, none would be statistically significant. Importantly, although there was variability of the values of rate and offsets estimated by each of the simulations, the differences between the estimates of rate and offset were small calculated by each program.

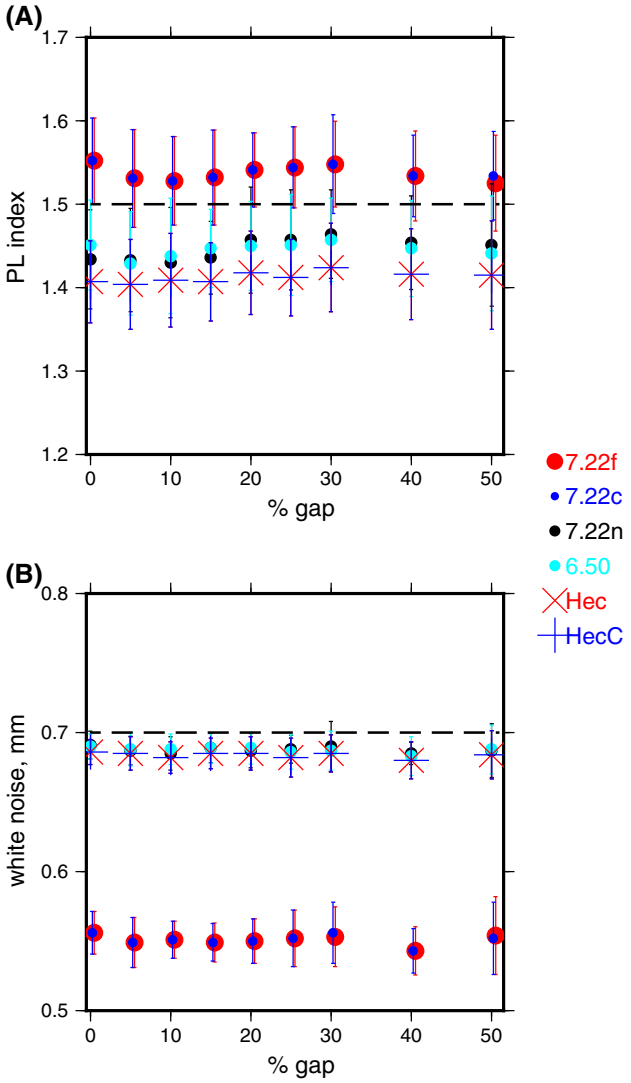

Fig. 2 Estimate of power-law index and white noise component from the six versions of maximum likelihood codes that estimate the optimal power-law noise representing the simulations in Fig. 1. All 15 simulations used power-law noise with an index of 1.5 and 0.7 mm white noise added in quadrature. In (**a**), the estimates of power-law index are shown, while in (**b**), the estimates of the white noise amplitude are shown. The *vertical bars* represent the 25–75% interval of either the index or white noise estimate. The *dashed*, *horizontal line* represents the simulated value with noise added in quadrature

4 Discussion

With the combination of reformulating the construction of the data error model from quadrature addition to simple addition of filters and reformulating the inverse of the covariance matrix provided by Bos et al. (2013), the algorithm discussed here has nearly the same computational speed as that implemented by Bos et al. (2013), and, importantly, has no restriction on power-law index. The key improvement over Bos et al. (2013) is the reformulation of the data error in terms of simple addition rather than quadrature addition.

Figures 1, 2a, and 3 in this report essentially reproduce Figs. 2 through 5 in Bos et al. (2013). One difference, which applies to Fig. 1, is that they compared their algorithm to the *CATS* program of Williams (2008). For comparison of speed of computation, Bos et al. (2013) show that the speed of *Hec* is roughly equivalent to *CATS* when the number of gaps in the time series is approximately 50%. I believe that this may not be a valid comparison since *CATS* may have not been optimized for speed; both versions of *est_noise* and *Hector* have been optimized. Perhaps more valid is a comparison of the two modes of both *est_noise7.22* and *Hector*. In contrast to the 50% value, the results shown in Fig. 1 show near equivalence in speed when the number of gaps is between 20 and 25% for both modes of *Hector*, with one using Cholesky decomposition similar to CATS and the second mode using the covariance decomposition and fast Toeplitz solver.

CPU speed for all of these programs scales with the number of data, n, and the number of gaps, m, where n is taken to be number of data constituting the time series without gaps. For standard Cholesky decomposition used in modes *7.22c*, *7.22n* and *HecC*, CPU time scales approximately with

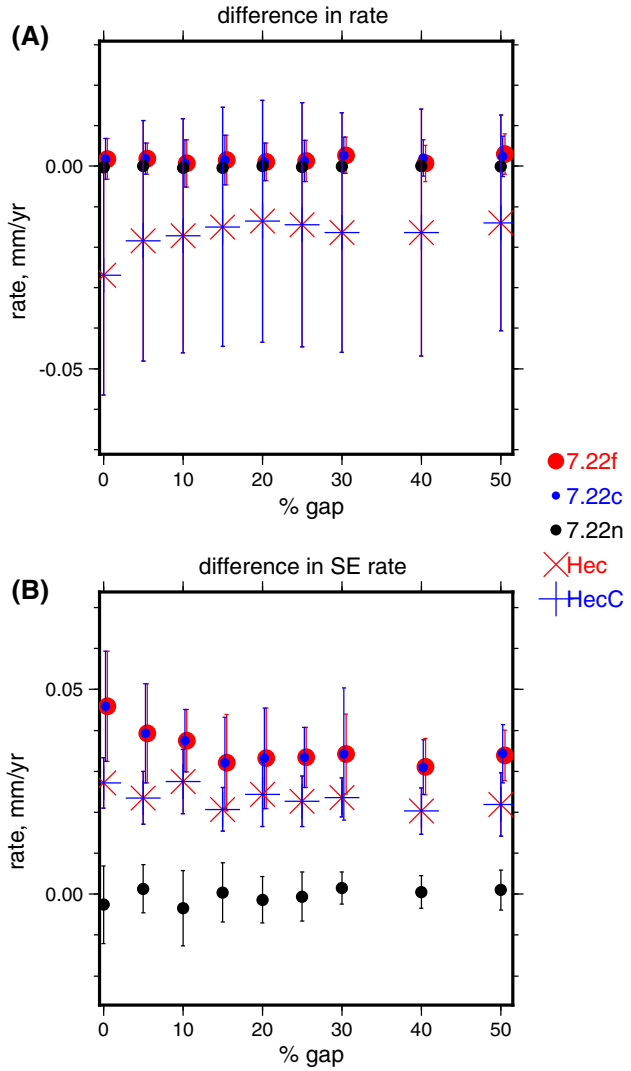

Fig. 3 Statistics of estimating rate and its uncertainty using three modes of *est_noise7.22* and two modes of *Hector* compared with the legacy *est_noise6.50*. The ordinate is the difference between the five estimates and that from *est_noise6.50*. In (**a**), the difference in estimated rate is shown while in (**b**), the difference in the standard error in rate is shown. The *vertical bars* represent the 25–75 % interval of either the estimate of the rate or its standard error

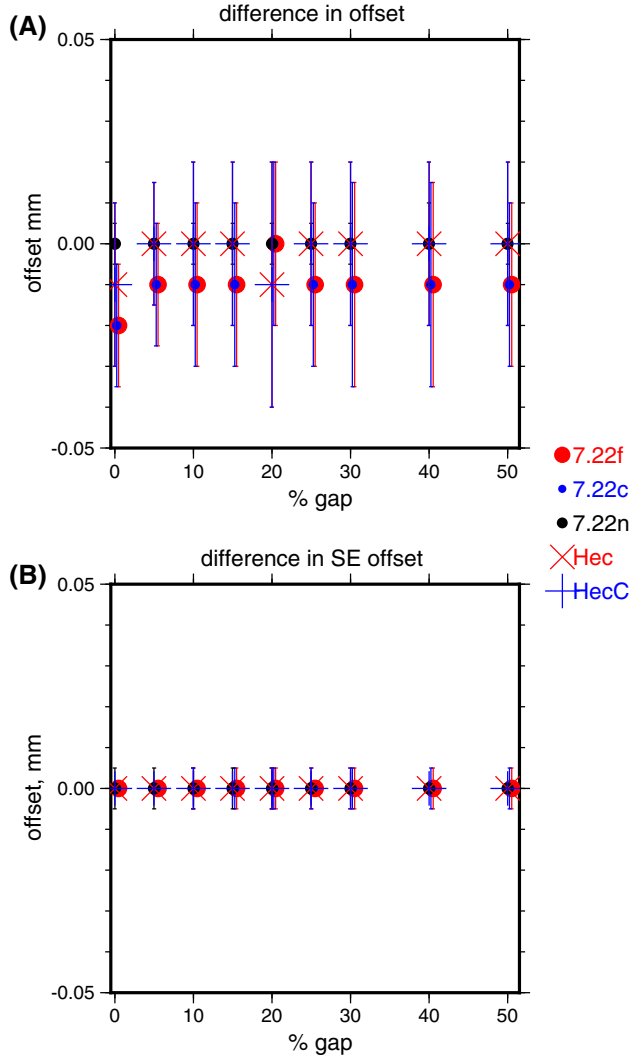

Fig. 4 Statistics of estimating offset and its uncertainty using three modes of *est_noise7.22* and two modes of *Hector* compared with the legacy *est_noise6.50*. The ordinate is the difference between the five estimates and that from *est_noise6.50*. In (**a**), the difference in estimated offset is shown while in (**b**), the difference in the standard error in offset is shown. The *vertical bars* represent the 25–75 % interval of either the estimate of the offset or its standard error; if there is *no bar*, then the length of the *bar* is less than the size of the symbol

$(n - m)^3$. Empirical tests suggest that the exponent ranges between 2.6 and 2.8.

On the other hand, the CPU scaling with using the Bos et al. (2013) reformulation of the data covariance is comprised of two components, one relating to the size of the covariance with no gaps and the other relating to the number of missing data. For the first part, CPU speed scales in n^2 for both mode *Hec*, using the fast Toeplitz solver and *7.22f* using the combination of deconvolution of the data noise, Eq. (9), and convolution of the filter with the data and model, Eq. (21). For the second component which involves both the n and m and for which Cholesky decomposition is required, CPU speed scales approximately as $n^\kappa \times m^\lambda$, where both κ and λ differ between the two algorithms. For *Hec*, the scaling is $n^1 \times m^{1.5}$, and for *7.22f*, the scaling is $n^{1.6} \times m^{1.2}$; these scaling relations are based upon fitting $n^\kappa \times m^\lambda$ to a series of simulations with n between 1000 and 5000, and the percentage of gaps between 5 and 50% for both programs.

The payoff of using the faster algorithms comes with analyzing data from large GNSS networks. For instance, the US Geological Survey monitors the displacements of many sites in the western US. For one network consisting of 180 sites spanning the San Francisco Bay area, the empirical relations derived from the limited speed tests using simulated data suggest that the algorithm that uses only Cholesky decom-

position to invert the covariance matrix, *7.22c* requires 37 hours. In contrast, using *7.22f*, requires 4.5 hours, or a factor of 8 speed-up. For this network, the median percentage of gaps is 2.2% with a 9.1 year median length.

For a prescribed noise model that is a mix between white noise and power-law noise, the estimates of these parameters will be slightly different depending upon whether the noise model is built in quadrature or as simple sums. This difference is illustrated in Fig. 5, but can be seen in the difference between the two example covariance matrices, equations 6 and 8. The transformation between the two covariance matrices is the addition of the crossterm ab where a is the amplitude of white noise and b is the amplitude of power-law noise. To construct the spectrum for quadrature addition of noise, shown in red in Fig. 5, it is simply a sum of $P_{\rm pl}/f^n$ and $P_{\rm wn}$, where $P_{\rm pl}/f^n$ is related to the power-law amplitude, b, (Langbein 2004, Eq. 11), and $P_{\rm wn} = a^2/f_{\rm ny}$, with $f_{\rm ny}$ being the Nyquist frequency.

However, computing the equivalent spectrum for noise constructed with simple addition of their underlying functions is difficult. The most expedient way to construct the spectrum is to employ a discrete Fourier transform (DFT) of summed filter function, Eq. (7). The results of this calculation, using the same parameters as those in red, are shown in blue. The differences between these two types of noise models can be most significant at the higher frequencies that represent the white noise component. Both of these noise models used 0.7 mm of white noise; yet, summed noise yields approximately 2 db more power than noise added in quadrature. Again, the root cause is the introduction of crossterms in the covariance matrix for simple sums.

In part, the scaling, especially for the white noise component between two methods of construction of the noise, explains the apparent difference in the white noise estimated that is shown in Fig. 2b. For the programs that construct the data covariance using quadrature addition, the estimated white noise averages to 0.69 mm within 0.01 mm of the simulated noise. However, the white noise amplitude averages to be 0.56 mm for the programs that use simple sums to construct the noise. However, since the diagonal terms of the data covariance for the simple sums are a factor of $2ab$ greater than that constructed with quadrature addition, I estimate the effective white noise to average 0.67 mm, or close to simulated white noise. In detail, to calculate the effective white noise requires computing the PSD using a DFT described below.

Reconciliation between the two methods of constructing the noise model is shown in Fig. 6. Here, from each simulation and its corresponding estimate of noise parameters, an equivalent power spectral density (PSD) is computed. For the noise parameters that use quadrature addition, the equivalent spectrum is $P_{\rm wh} + P_{\rm pl}/f^{\alpha}$. However, to find the equivalent PSD for the noise parameters obtained with the assumption

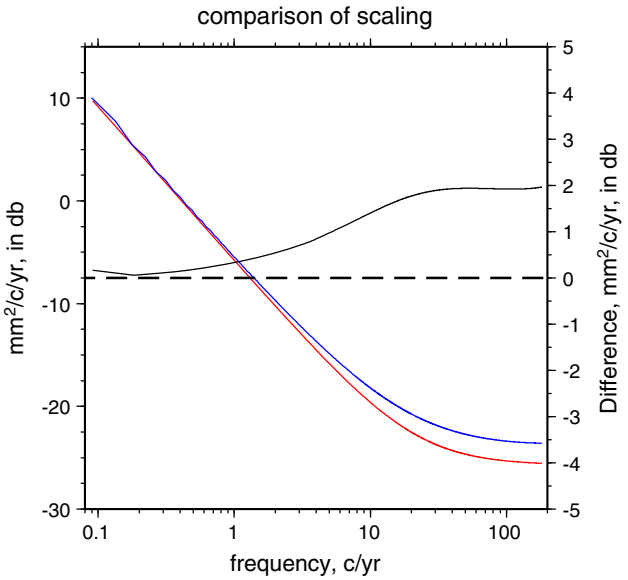

Fig. 5 Underlying power spectra for a noise model consisting of power law and white noise. The power law has an index of 1.5 and an amplitude of 3 mm/year$^{0.375}$ and the white noise is 0.7 mm. Plotted in *red* is the spectrum when these two components are added in quadrature. *Blue* is the spectrum when the two components are from simple addition of their filters using the same values as used for quadrature addition. The difference between the two is shown in black with its ordinate scale shown on the *right-hand side*. The simulation discussed in the text uses quadrature addition

that noise is additive, a DFT is used. Since 15 simulations of noise were used, the PSDs shown in Fig. 6 are the median spectra from each mode, *7.22n* and *7.22f*. In spite of large, apparent spread in the estimates of the white noise parameter, Fig. 2b, the spectra representing the two modes of computing data covariance are nearly equivalent.

The comparison of the two PSDs in Fig. 6 shows some divergence at the low frequencies, with the mode that uses additive noise having slightly more power. This is reflected in the estimates of standard error for rate, where those standard errors are larger when the covariance is constructed from additive noise. However, as mentioned previously, the differences are slight, at 10%.

Fundamentally, the choice is arbitrary whether to use noise added in quadrature or by simple addition of filter functions. Essentially, either approach, as shown in Figs. 3 and 4, yields similar estimates of the model that describes the time dependence of the data. The chief difference is the potential for rapid computation simultaneously of both the noise- and time-dependent models provided by simple addition with the potential for misinterpretation of the white noise amplitude. Instead, the white noise amplitude provided directly from the additive noise model needs to added to the high-frequency portion of the temporal covariance spectra through the DFT.

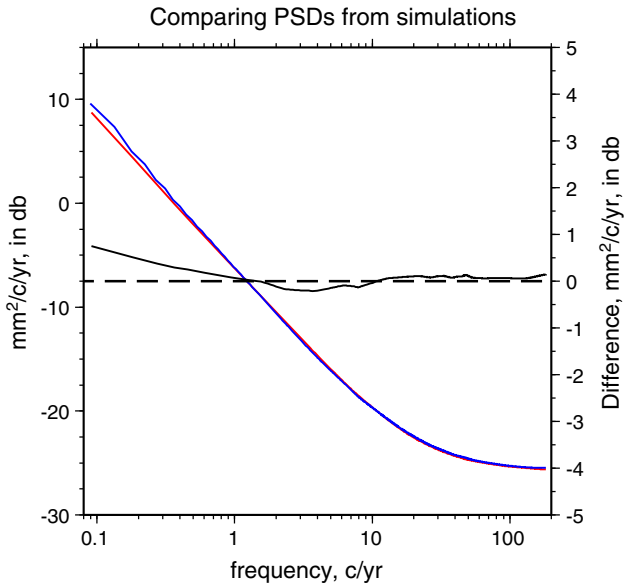

Fig. 6 Comparisons of the power spectral densities from the simulations derived from noise estimates from modes *7.22n* and *7.22f*. Plotted in *red* is the spectrum derived from *7.22n* where data covariance assumed that the noise is added in quadrature. Plotted in *blue* is the spectrum derived from *7.22f*, where data covariance assumed that the noise is from simple addition. The difference between the two is shown in black, with its ordinate scale shown on the *right-hand side*. Near equivalence is achieved since the white noise amplitude from *7.22f* averages 0.56 mm and the white noise from *7.22n* averages 0.69 mm

Although both this work and the work of Bos et al. (2013) using maximum likelihood methods provide improved efficiencies working with long time series, one needs to be judicious using these algorithms over other techniques. For instance, over the past decade, it is now common to work with GNSS data sampled at 1 sample per second (sps) rather than daily estimates of position. This represents almost a factor of 10^5 more data and, for *est_noise7.22*, the number of data could overwhelm both the array dimensions and computer memory. Consequently, standard power spectral techniques that revolve around DFT, windowing, and averaging should be considered and employed.

For example, with high-rate GNSS data, most of the data will only record background noise which has significant temporal correlations (Langbein and Bock 2004; Genrich and Bock 2006). From those records, the power spectrum can be estimated, then generalized by either graphically fitting a power-law and white noise function to the spectrum or using a more sophisticated, but unspecified method. That noise model, with aid of scaling provided by Langbein (2008), Eq. (11), can be used as an input to *est_noise7.22* on an interval of high-rate data that exhibits transient deformation for which there is a standard function that could represent the size of the transient. This is a very fast calculation as the noise model is known and is held fixed.

5 Conclusions

The new algorithm that merges a different method of modeling temporal correlation in data and the Bos et al. (2013) method of partitioning the data covariance matrix provides a rapid computation estimating both the parameters that describe a time-dependent function that underlies the data and the data error that can provide realistic estimates of the uncertainties of the parameters of the time dependent function. Bos et al. (2013) describe an approximation of the data covariance, that is a Toeplitz matrix, which allows for rapid inversion of the large data covariance. However, that approximation can be restrictive and contrary to data which have large temporal covariance, in particular any data that have a power-law index >1.6, which includes random walk. Further extension of the approximation uses generalized Gauss–Markov noise to allow the power-law index to exceed 1.6.

That approximation is eliminated by making a different assumption with respect to how the data noise is constructed. Rather than noise being constructed from independent noise sources, the noise is constructed from a single source and convolved with a complex filter that can comprise white, power-law, and/or bandpass-filtered components. Consequently, this results in a rapid method of inverting the data covariance matrix reducing an n^3 calculation to an n^2 calculation.

For the simulations discussed here, where the power-law index was taken to be 1.5, both methods provide equivalent results.

Acknowledgements I would like thank Jessica Murray and Emily Montgomery-Brown for their critical reviews. Discussions over the past 10 years with Machiel Bos and Simon Williams contributed to this work.

Appendix 1: More efficient algorithm

The efficiency of evaluating the Bos et al. (2013), Eq. (7) can be improved by not directly computing C^{-1} for the case where the covariance is constructed by simple sums. Instead, using convolution yields up to a factor of 5 increase in computation speed when there are no missing data. Starting with the covariance adjustment portion of Eq. (12),

$$C_{\text{adj}}^{-1} = C^{-1} - C^{-1}M(M^{\text{t}}C^{-1}M)^{-1}M^{\text{t}}C^{-1} \tag{14}$$

this can be plugged into

$$\breve{x} = [A^{\text{t}}C_{\text{adj}}^{-1}A]^{-1}A^{\text{t}}C_{\text{adj}}^{-1}d \tag{15}$$

where A is the observation matrix with n rows with zeros substituted at times of missing observations. Likewise, d is an n length data vector with zeros at times of missing data.

Furthermore, the function, f^{-1}, Eq. (9), can be considered as a filter, which if chosen correctly, and convolved with the data, will whiten those data; $d(t) * f^{-1}$. That is, rather than $d(t)$ having a power-law power spectrum, the convolved spectrum becomes "flat" or frequency independent representing white noise. The matrix F^{-1} comprising f^{-1} will be identified for convenience as W, or

$$C^{-1} = W^{\mathrm{t}} W \tag{16}$$

Both Eqs. (14) and (16) can be plugged into Eq. (15). Using the following substitutions to regroup the terms;

$$d_{\mathrm{w}} = Wd \tag{17}$$

$$A_{\mathrm{w}} = WA \tag{18}$$

this will "whiten" both the data and the observation matrix. In addition, the filter on the missing data is formed by

$$E = WM \tag{19}$$

Recall that the matrix multiplication using M is simply selecting the columns in W representing missing observations.

Making the above substitutions yields

$$\begin{aligned}\check{x} = &[A_{\mathrm{w}}^{\mathrm{t}} A_{\mathrm{w}} - A_{\mathrm{w}}^{\mathrm{t}} E (E^{\mathrm{t}} E)^{-1} E^{\mathrm{t}} A_{\mathrm{w}}]^{-1} \\ &[A_{\mathrm{w}}^{\mathrm{t}} - A_{\mathrm{w}}^{\mathrm{t}} E (E^{\mathrm{t}} E)^{-1} E^{\mathrm{t}}] d_{\mathrm{w}}\end{aligned} \tag{20}$$

And, substituting $B = E(E^{\mathrm{t}} E)^{-1} E^{\mathrm{t}}$ yields

$$\check{x} = [A_{\mathrm{w}}^{\mathrm{t}} A_{\mathrm{w}} - A_{\mathrm{w}}^{\mathrm{t}} B A_{\mathrm{w}}]^{-1} [A_{\mathrm{w}}^{\mathrm{t}} - A_{\mathrm{w}}^{\mathrm{t}} B] d_{\mathrm{w}} \tag{21}$$

Using Eq. (21) rather than the combination of Eqs. (14) and (15) provides some computational efficiency using convolution and BLAS computer programs to compute $A_{\mathrm{w}}^{\mathrm{t}} A_{\mathrm{w}}$. One gets roughly a factor of five speed-up in the case where there are no missing observations. Once the number of missing observations becomes significant, the computation of both B and its surrounding terms that involve A_{w} becomes significant, but no worse than the calculation involving both Eqs. (14) and (15).

Appendix 2: Source code

A copy of the source code and other documentation can be found at http://earthquake.usgs.gov/research/software/#est_noise.

References

Agnew DC (1992) The time domain behavior of power law noises. Geophys Res Lett. doi:10.1029/91GL02832

Amiri-Simkooei AR, Tiberius CCJM, Teunissen PJG (2007) Assessment of noise in GPS coordinate time series: methodology and results. J Geophys Res. doi:10.1029/2006JB004913

Bos MS, Fernandes RMS, Williams SPD, Bastos L (2008) Fast error analysis of continuous GPS observations. J Geod. doi:10.1007/s00190-007-0165-x

Bos MS, Fernandes RMS, Williams SPD, Bastos L (2013) Fast error analysis of continuous GNSS observations with missing data. J Geod. doi:10.1007/s00190-012-0605-0

Bos M, Fernandes R (2015) Investigation of random-walk noise in GNSS time-series, *Am Geophys* Union, Abstracts, Fall 2015 meeting. https://agu.confex.com/agu/fm15/meetingapp.cgi/Paper/73901

Dmitrieva K, Segall P, DeMets C (2015) Network-based estimation of time-dependent noise in GPS position time series. J Geod 89(6):591–606. doi:10.1007/s00190-015-0801-9

Genrich JF, Bock Y (2006) Instantaneous geodetic positioning with 10–50 Hz GPS measurements: Noise characteristics and implications for monitoring networks. J Geophys Res. doi:10.1029/2005JB003617

Hackl M, Malservisi R, Hugentobler U, Wonnacott R (2011) Estimation of velocity uncertainties from GPS time series: examples from analysis of the South African TrigNet network. J Geophys Res 116B15:11404

Hosking JRM (1981) Fractional differences. Biometrika 68:165–176

Kasdin NJ (1995) Discrete simulation of colored noise and stochastic processes and $1/f^{\alpha}$ power-law noise generation. Proc IEEE 83(5):802–827

Langbein J, Johnson H (1997) Correlated error in geodetic time series: implications for time-dependent deformation. J Geophys Res 102:591–604

Langbein J, Bock Y (2004) High-rate real-time GPS network at Parkfield: utility for detecting fault slip and seismic displacements. Geophys Res Lett 31:L15S20. doi:10.1029/2003GL019408

Langbein J (2004) Noise in two-color electronic distance meter measurements revisited. J Geophys Res. doi:10.1029/2003JB002819

Langbein J (2008) Noise in GPS displacement measurements from Southern California and Southern Nevada. J Geophys Res. doi:10.1029/2007JB005247

Langbein J (2012) Estimating rate uncertainty with maximum likelihood: differences between power-law and flicker random-walk models. J Geod 86(9):775–783. doi:10.1007/s00190-012-0556-5

Mao A, Harrison CGA, Dixon TH (1999) Noise in GPS coordinate time series. J Geophys Res 104:2797–2816

Nelder JA, Mead R (1965) A simplex method for function minimization. Comput J 7:308–313. doi:10.1093/comjnl/7.4.308

Williams SDP (2003) The effect of coloured noise on the of rates from geodetic time series. J Geod 76(9–10):483–494. doi:10.1007/s00190-002-283-4

Williams SDP (2008) CATS: GPS coordinate time series analysis software. GPS Solut 12(2):147–153. doi:10.1007/s10291-007-0086-4

Williams SDP, Bock Y, Fang P, Jamason P, Nikolaidis RM, Prawirodirdjo L, Miller M, Johnson DJ (2004) Error analysis of continuous GPS position time series. J Geophys Res. doi:10.1029/2003JB002741