IVS Memorandum 2006-013v03

4 September 2007

"Simulation of wet zenith delays and clocks"

Johannes Böhm, Jörg Wresnik, Andrea Pany

Memo: Simulation of wet delays and clocks
Authors: Johannes Böhm, Jörg Wresnik, and Andrea Pany
Date: 4 September 2007

Introduction

With the VLBI2010 Monte Carlo simulations we do not have access to the o-c vector (observed minus computed) for the least-squares adjustments. Thus, we have to set up these vectors with simulated data:

(1) $o\text{-}c = (wzd_2 \,.\, mfw_2(e) + clock_2) - (wzd_1 \,.\, mfw_1(e) + clock_1) + wn$

$wzd_{1,2}$ and $clock_{1,2}$ are the simulated wet zenith delays and clock values at the first and second station, and $mfw_{1,2}(e)$ are the corresponding wet mapping functions for the elevation angles e. wn is the white noise that accounts for thermal noise, and it is added per baseline. The default value for the standard deviation of the white noise wn for VLBI2010 is 4 psec.

Simulation of wet delays

In order to simulate the wet delays as realistically as possible we apply turbulence theory (Treuhaft and Lanyi 1987). In particular, we follow the strategy proposed by Nilsson et al. (2007).

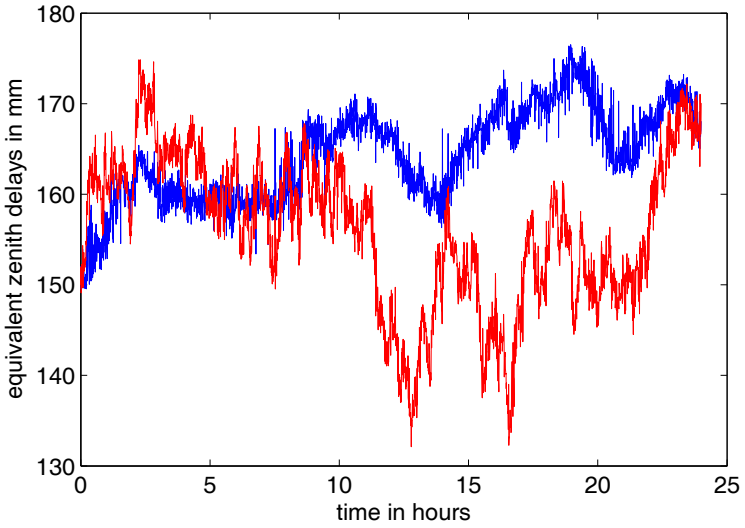

Figure 1. Equivalent wet zenith delays from the turbulence model with the following parameters: h = 1 km, $Cn = 2.4 \cdot 10^{-7}$ $m^{-1/3}$; blue: wind is 1 m/s towards east, red: wind is 8 m/s towards east.

Unlike simulations with random walk or first-order Gauss Markov processes for the wet zenith delays (Herring et al. 1990), the wet delays from turbulence theory also include the azimuthal variation which is taken into account with the covariance information between all observations at a station. The output of these calculations is a time series of equivalent zenith delays that include the dependency on elevation, azimuth and time epoch (w.r.t. the first epoch) of the observation.

There are several parameters that are very important when creating the turbulent delays. These are
1) the initial wet zenith delay,
2) the wind velocity,
3) the Cn parameter, i.e. structure constant,
4) the effective height,
5) and the height increment for the numerical integration.
There should be agreement which values are to be used for which stations and so forth. In the Appendix the Matlab source code is provided. It is based on Nilsson et al. (2007).

Simulation of clocks

The clocks are set up as the sum of a random walk process and an integrated random walk process both being excited randomly. The VLBI2010 default value for the Allan Standard Deviation (ASD) is $2 \cdot 10^{-15}$ at 50 minutes. For more information concerning the close we refer to Herring et al. (1990).

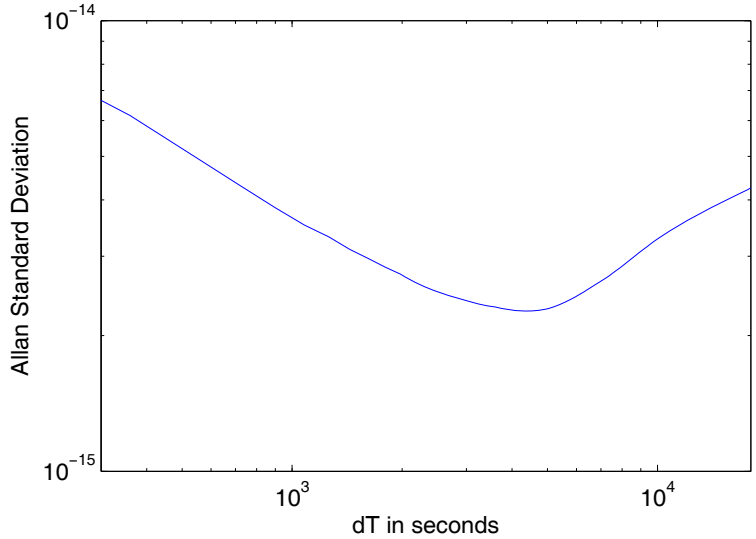

Figure 2. Allan Standard Deviation as derived for a simulated clock with an ASD of $2 \cdot 10^{-15}$ at 50 minutes.

References

Herring, T.A., J.L. Davis, and I.I. Shapiro, Geodesy by Radio Interferometry: The Application of Kalman Filtering to the Analysis of Very Long Baseline Interferometry Data, Journal of Geophysical Research, Volume 95, No. B8, pp 12561-12581, 1990.

Nilsson, T., R. Haas, and G. Elgered, Simulations of atmospheric path delays using turbulence models, Proceedings of the 18th European VLBI for Geodesy and Astrometry Working Meeting, 12-13 April 2007, edited by J. Boehm, A. Pany, and H. Schuh, Geowissenschaftliche Mitteilungen, Heft Nr. 79, Schriftenreihe der Studienrichtung Vermessung und Geoinformation, Technische Universitaet Wien, ISSN 1811-8380, 2007.

Treuhaft, R.N., and G.E. Lanyi, The effect of the dynamic wet troposphere on radio interferometric measurements, Radio Science, Volume 22, No. 2, pp 251-265, 1987.

Appendix A: Simulation of clocks

```
function [tt,tc] = predictclock_rwirw;
% tt:  time in seconds
% tc:  clock readings in seconds
% noise level of the clock
SY = 1e-14;  % sec/sec
dt = 1;      % 1 second time interval
cn = 0;
phir = SY^2*50*60;     % sec^2/sec
phii = SY^2/(50*60)*3; % sec^2/sec^3
wnsr = sqrt(phir);     % sec/sqrt(sec)
wnsi = sqrt(phii);     % sec/(sec*sqrt(sec))
% Generate the time series
num = 86400/dt + 1;
ti = zeros(num,1);
tr = zeros(num,1);
tt = [1:num]*dt-dt;   % Times of values
% random walk
% ---
tr(1) = randn*cn;
y = tr(1);
for t = 2:length(tt);
    wn = randn*wnsr/sqrt(dt);  % sec/sec
    y = y + wn*dt + randn*cn;  % sec
    tr(t) = y;
end
% integrated random walk
% ---
ti(1) = randn*cn;
y = ti(1);
v = 0;
for t = 2:length(tt)
    wn = randn*wnsi/sqrt(dt);  % sec/sec^2
    y = y + v*dt - (wn)/2*dt^2 + randn*cn;
    v = v + wn*dt;    % sec/sec
    ti(t) = y;
end
% sum of random walk and integrated random walk
tc = tr + ti;
```

Appendix B: Simulation of equivalent wet zenith delays

```
% constants
Cn    = (2.4e-7);          % [m^(-1/3)]
L0    = 3e6;               % [m] saturation scale length
vv    = [0 1*3600 0];      % [m/h] wind in East direction
h     = 1000;              % [m] effective height of troposphere
dh    = 200;
%
t   = [0:5/60:24-5/60];    % example: every 5 min for 24 hours
num = length(t);   % including reference epoch
for i = 1:num
    az(i) = rand*2*pi;
    el(i) = (pi/2-rand*85*pi/180);
    ri(i,1:3) = [cos(az(i))/tan(el(i)) sin(az(i))/tan(el(i)) 1];
end
%
az1 = 0;
el1 = pi/2;
r0 = [cos(az1)/tan(el1) sin(az1)/tan(el1) 1];
%
k = 0;
for i = 1:6
    for j = 1:6
        k = k + 1;
        zs(k) = dh*(i-1);
        z (k) = dh*(j-1);
        v(1:3,k) = vv';
    end
end
%
r0z  = r0'*z;
r0zs = r0'*zs;
rho4 = sqrt(sum((r0z-r0zs).^2));
%
for i = 2:num
    if mod(i,100)==0
      i
    end
    riz  = ri(i,1:3)'*z;
    dti0 = t(i);
    rho1 = sqrt(sum((riz-r0zs+v*dti0).^2));
    for j = i:num
        % compute time differences in hours
        dtj0 = t(j);
        dtij = t(j)-t(i);
        % compute position vectors
        rjz  = ri(j,1:3)'*z;
        rjzs = ri(j,1:3)'*zs;
        % compute separations
        rho2 = sqrt(sum((rjz-r0zs+v*dtj0).^2));
        rho3 = sqrt(sum((riz-rjzs-v*dtij).^2));
        out  = rho1.^(2/3)./(1+rho1.^(2/3)/L0^(2/3)) + ...
               rho2.^(2/3)./(1+rho2.^(2/3)/L0^(2/3)) - ...
               rho3.^(2/3)./(1+rho3.^(2/3)/L0^(2/3)) - ...
               rho4.^(2/3)./(1+rho4.^(2/3)/L0^(2/3));%
        result = sum(out)*dh^2;
        C(i,j) = Cn^2/2*1e6*result;
  end
end
% delete column and row for reference delay l_0
C(1,:) = [];
```

```
C(:,1) = [];
% 'inv'
D = chol(C);
x = randn(num,1);
x(1) = [];
wzd = 150 + D'*x;    % 150 mm is the wet zenith delay at the first epoch
```