LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY
- LIGO -
CALIFORNIA INSTITUTE OF TECHNOLOGY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Technical Note	**LIGO-T070025-00-Z**	2007-02-06

Synchronization issues affecting the fsr stochastic search during S4

T. Fricke, S. Giampanis and A. C. Melissinos
University of Rochester Stochastic Analysis Group

Distribution of this document:

Stochastic Analysis Group

Draft
Circulation restricted to LIGO I members

This is an internal working
note of the LIGO project

California Institute of Technology
LIGO Project, MS 18-34
Pasadena, CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project, Room NW17-161
Cambridge, MA 02139
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

LIGO Hanford Observatory
Route 10, Mile Marker 2
Richland, WA 99352
Phone (509) 372-8106
Fax (509) 372-8137
E-mail: info@ligo.caltech.edu

LIGO Livingston Observatory
19100 LIGO Lane
Livingston, LA 70754
Phone (225) 686-3100
Fax (225) 686-7189
E-mail: info@ligo.caltech.edu

WWW: http://www.ligo.caltech.edu/

Abstract

This is a brief description of timing/synchronization issues affecting the S4 FSR stochastic search

1 Introduction

A delay in the relative timing between two time series obviously modifies their cross correlation. Suppose

$$h_1(t) = h(t) \qquad \text{and} \qquad h_2(t) = h(t+\tau)$$

Taking the Fourier transforms

$$\begin{aligned} x_1(f) &= s(f) + n_1(f) \\ x_2(f) &= s(f)e^{2\pi i f\tau} + n_2(f) \end{aligned}$$

Here $s(f)$ is the signal common to both time series and $n_1(f)$, $n_2(f)$ are the noise in the respective channels (we ignore the effect of the time shift on the noise). The cross-correlation in the frequency domain is given by

$$\langle x_1^*(f)x_2(f)\rangle = |s(f)|^2 e^{2\pi i f\tau} \tag{1}$$

The "statistic" that we seek acquires a frequency dependent phase.

Given the expected values of τ, this extra phase is negligible for low frequency measurements but must be accounted for in the fsr regions. We distinguish three possible timing effects.

1. A static time delay τ that may exist between the two time series.
2. Fluctuations of the time delay about a fixed value of τ. We refer to this as the "jitter" $\delta\tau$.
3. The static time delay may change at discrete points during the run, such as when the DAQ system is rebooted.

The analysis of the fsr data is carried out over a narrow frequency band ($\Delta f = \pm 200$ Hz at $f = 37.52$ and $\Delta f = \pm 280$ Hz at $f = 75.0$ kHz. Thus the phase-shift in Eq(1) can be evaluated at the central frequency and used over the entire band.

2 Timing Jitter

Let the real cross-correlation (with phase $\theta = 0$) have the value Y_0. When the phase angle is θ, the measured cross-correlation Y, has the value $Y = Y_0 \cos\theta$. The phase θ is Gaussian distributed around $\theta = 0$ with $\sigma_\theta = 2\pi f(\delta\tau)$. Thus

$$\langle Y\rangle/Y_0 = \langle\cos\theta\rangle = (\int_{-\pi}^{\pi} \cos\theta e^{-\theta^2/2\sigma_\theta^2} d\theta)/(\int_{-\pi}^{\pi} e^{-\theta^2/2\sigma_\theta^2}) = N(\sigma_\theta)e^{-\sigma_\theta^2/2} \tag{2}$$

The measured statistic is smaller than the true correlation by the exponential in the last term of Eq (2)multiplied by the normalization factor $N(\sigma_\theta)$. This factor is equal to unity for values of $\sigma_\theta < 1$ rad.

As shown in Appendix I we can evaluate the time jitter from the data and find

$$\delta\tau = (0.7 \pm 1)\mu s$$

Using $\tau_0 = 2\pi f\langle\delta\tau\rangle$ we find the following corrections to the measured statistic

σ_θ	0.165	0.330 radians
$Y_0/\langle Y\rangle$	1.014	1.056

These results have also been verified by a numerical simulation.

3 Static Time Delay

An arbitrary static delay τ, adds a phase $\theta = 2\pi f\tau$ to the statistic, $0 \le \theta < 2\pi$. As before let Y_0 be the (true) value of the statistic when $\theta = 0$. We measure instead the two components

$$\begin{aligned} x &= Y_0\cos\theta \qquad \text{(real part)} \\ y &= Y_0\sin\theta \qquad \text{(imaginary part)} \end{aligned}$$

with a Gaussian error σ. The joint probability of obtaining (x, y) given Y_0, θ and σ is

$$\begin{aligned} P(x, y|Y_0, \theta, \sigma) &\propto \exp\left[-(Y_0\cos\theta - x)^2/2\sigma^2\right]\exp\left[-(Y_0\sin\theta - y)^2/2\sigma^2\right] \\ &= \exp\left[-(Y_0 + Y^2)/2\sigma^2\right]\exp\left[Y_0(x\cos\theta + y\sin\theta)/\sigma^2\right] \end{aligned} \tag{3}$$

where we have set $x^2 + y^2 = Y^2$. Eq (3) can be simplified by noting that

$$x = Y\cos\phi \qquad y = Y\sin\phi$$

with $Y = \sqrt{Y^2}$ and $\tan\phi = y/x$. It then follows

$$x\cos\theta + y\sin\theta = Y\cos(\theta - \phi).$$

Introducing this expression in Eq (3) allows us to integrate over θ analytically

$$\begin{aligned} P(Y|Y_0,\sigma) &= N\int_0^{2\pi} P(Y|Y_0,\theta,\sigma)d\theta \\ &= N\exp\left[-(Y_0^2+Y^2)/2\sigma^2\right] I_0(Y_0Y/\sigma^2) \end{aligned} \tag{4}$$

where $I_0(x)$ is the modified Bessel function of the first kind of order zero, and N the appropriate normalization constant.

Using Bayes' theorem the probability of any given value Y_0 is

$$P(Y_0|Y,\sigma) \propto P(Y|Y_0,\sigma)\pi(Y_0) \tag{5}$$

where $\pi(Y_0)$ is the prior probability for Y_0. We take $\pi(Y_0)$ to be flat from zero up to some positive upper limit. Eq (4) has been checked against a numerical integration of Eq (3) over θ.

4 Variable Time Delay

When we have many measurements Y_J, σ_J possibly obtained with a different time delay we proceed as follows. Calculate $P_J(Y_0|Y_J,\sigma_J)$ for each measurement using Eq (5) and form their product

$$\mathcal{L}(Y_0) \propto \prod_J P_J(Y_0|Y_J,\sigma_J) \tag{6}$$

This is the likelihood function for Y_0 given the set of measured values. Numerically we calculate the logarithm of Eq (6)

$$W(Y_0) = -\log[\mathcal{L}(Y_0] = \Sigma_J\left\{(Y_0^2+Y_J^2)/2\sigma_J^2 - \log[I_0(Y_0Y_J/\sigma_J^2)]\right\}$$

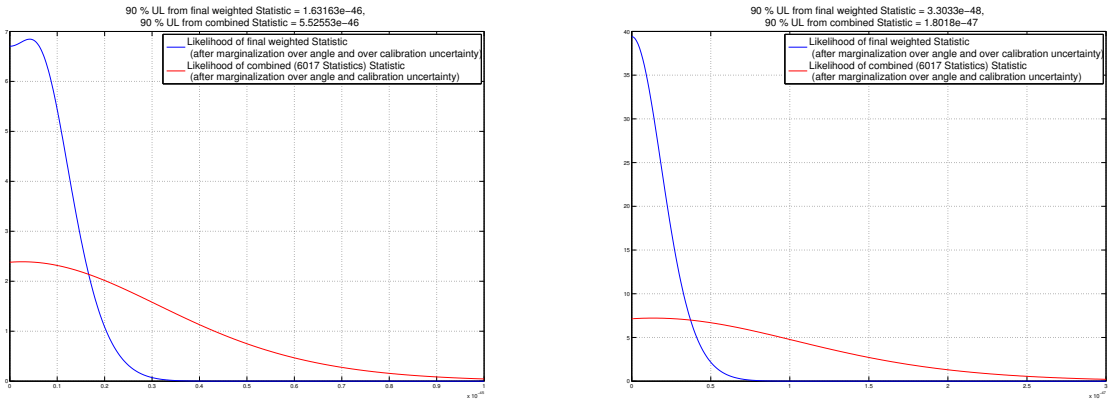
and then exponentiate

$$\mathcal{L}(Y_0) = Ce^{-W(Y_0)}$$

C is a normalization constant such that

$$\int_\alpha^\beta \mathcal{L}(Y_0)dY_0 = 1$$

Here α, β are the limits on the prior values of Y_0 assumed to be flat in that range.



(a) Likelihood functions for the 37.5 kHz region (b) Likelihood functions for the 75 kHz region

Figure 1: The likelihood functions for $|h_d|^2$. The blue curve is obtained by first averaging the 6017 statistic and then marginalizing over the unknown phase, as well as over the calibration uncertainty. The red curve is obtained by marginalizing each of the 6017 statistics over an unknown phase, and then forming the overall likelihood; the combined likelihood is then marginalized over the calibration uncertainty.

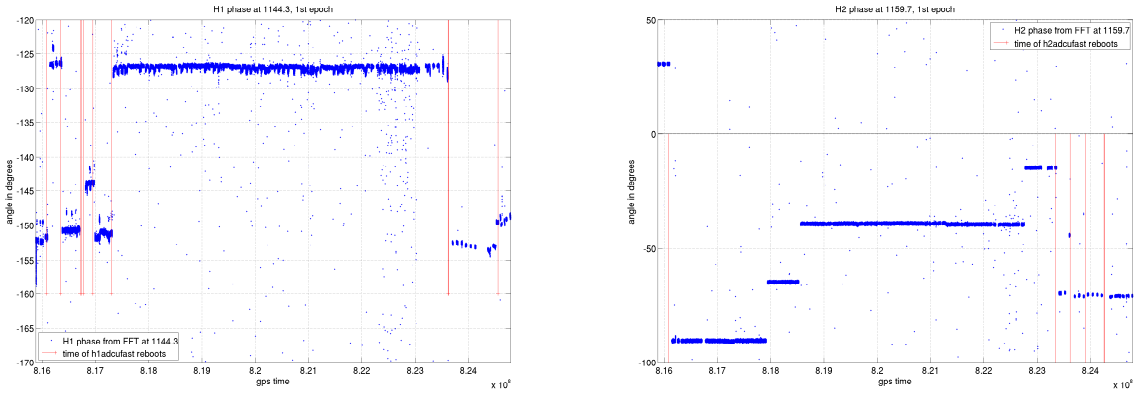
Typical results using the S4 data at 1 fsr are shown in Fig.1.

There were 6017 independent measurements, and their weighted average is

$$< x >= 1.11 \times 10^{-47} \qquad < y >= 9.33 \times 10^{-47} \qquad < \sigma >= 6.15 \times 10^{-47}$$

(the units are (strain)^2/Hz). Averaging over angles gives the posterior probability shown by the blue curve. If instead we first calculate the probability for each measurement and then form the overall likelihood we obtain the red curve. The analogous results for the S4 data at 2 fsr are shown in Fig.1.

The code used to obtain these results is given in Appendix II.



(a) First epoch H1 phase of calibration line at 1144.3 Hz (b) First epoch H2 phase of calibration line at 1159.7 Hz

Figure 2: Phases of calibration lines in H1 and H2.

5 APPENDIX I - Measurement of the time jitter

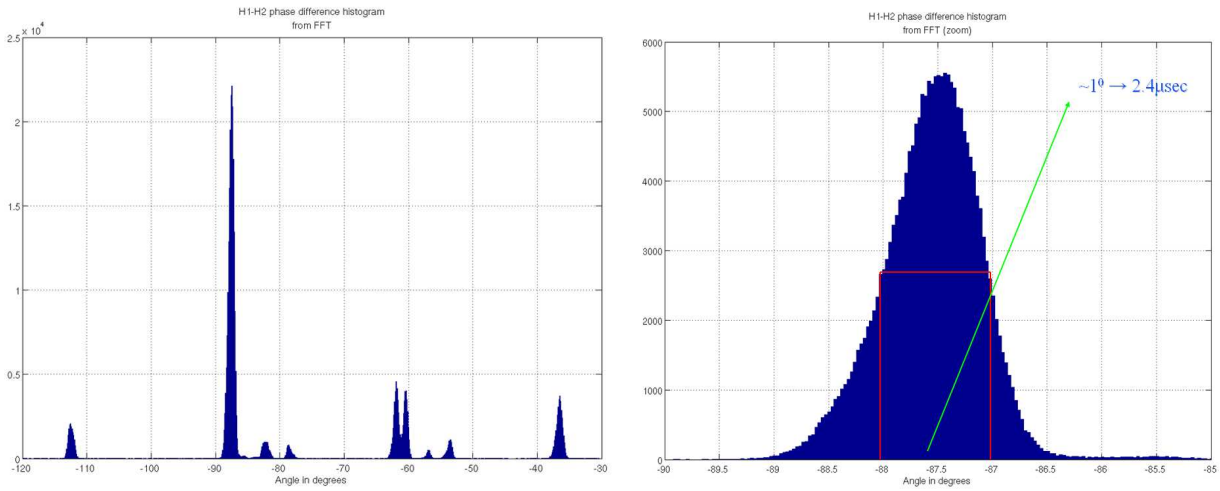
During S5 the fast channels were heterodyned to low frequency so that the range 0 - 2048 kHz was accessible in the recorded frames. This allows the continuous monitoring of the amplitude and phase of the (upper two) calibration lines:

We then form the difference in the phase of the lines measured for H1 and H2 as a function of time. This is shown in Fig.2 for the higher frequency line ($f \sim 1150$ Hz) for the period from November 13 2005 to February 24 2005 (first epoch of S5 data). There are distinct jumps in the phase difference and they correspond exactly with reboots of the DAQ system as recorded in the e-log.

Fig.3 is a histogram of the phase difference (for the $f \sim 1150$ Hz) line. There are several distinct peaks due to the different delays in the timing seen in Fig.2. The width of these peaks is a measure of the time jitter. Typically FWHM $\sim 1^0$, and thus $\sigma = FWHM/(2\sqrt{2ln2}) \sim 0.42^0$. Converting this phase difference to time (at $f = 1150$Hz) gives

$$\Delta\tau = \Delta\theta/(360 \cdot f) \sim 1\mu\text{sec}$$

However this is an upper limit because the determination of the individual phases is subject to measurement error. This error can be estimated by histrogramming the measured phase of a single line. It turns out that the width of the distribution of the phase of a single line is of the same order as in Fig.3.



(a) Histogram (b) zoom

Figure 3: Histogram of phase difference between 1144.3Hz in H1 and 1159.7 Hz in H2 calibration line

6 APPENDIX II - Code ("*marginalize.m*")

```
0001 function [varargout] = marginalize(out_path, varargin)
0002 % marginalize('','1fsr/analysis_opt.csv',10^(-44),1000,0);
0003
0004 infile    = varargin{1};  % data for 1 or 2 FSR
0005 priorMax  = varargin{2};  % maximum value of prior PDF
(eg. e-44 for 1fsr, e-45 for 2fsr);
0006 n         = varargin{3};  % number of points for prior PDF (flat)
0007 doMargCal = varargin{4};  % boolean (0/1) parameters for marginalization
over calibration uncertainty
0008
0009 %% Read data
0010 data = dlmread(infile);
0011
0012 %% Sort the data by gpstime
0013 data = sortrows(data, 5);
0014
0015 %% Pull out the fields of interest
0016 rstat   = data(:,2);
0017 istat   = data(:,3);
0018 sigma   = data(:,4);
0019 gpstime = data(:,5);
0020 alpha1  = data(:,6);
0021 alpha2  = data(:,7);
0022 ps1     = data(:,8);
0023 ps2     = data(:,9);
```

```
0024 days = (gpstime - min(gpstime)) / (60 * 60 * 24);
0025
0026 %% Confidence level
0027 confidence = 0.9;
0028
0029 %% Weighted average of Statistics
0030 fprintf('Computing weighted averages:\n');
0031 weighted_mean = @(x, w) (x' * w)/sum(w);
0032 rstat_wmean = weighted_mean(rstat, sigma.^(-2))
0033 istat_wmean = weighted_mean(istat, sigma.^(-2))
0034
0035 sigma_error = sum(sigma.^(-2))^(-1/2)
0036
0037 fprintf('frequentist 90%% upper limit from weighted Statistic:\n')
0038 if rstat_wmean < 0
0039     UL_f = 1.645 * sigma_error % 90% frequentist upper limit
0040     else
0041         UL_f = rstat_wmean + 1.645 * sigma_error
0042 end;
0043
0044 %% Marginalization over angle for final weighted statistic
0045 r0   = linspace(0,priorMax,n); % prior PDF (flat)
0046
0047 y_t  = sqrt(rstat_wmean^2 + istat_wmean^2); % magnitude of weighted Statistic
0048 y_t  = repmat(y_t,1,n);
0049 sigma2 = repmat(sigma_error.^2,1,n);
0050 p_y    = exp(- (y_t.^2 + r0.^2) ./ (2*(sigma2))) .* besseli(0,y_t.*r0./sigma2);
% posterior PDF
0051 likelihood_f = p_y / sum(p_y/n); % total likelihood normalized to integrate to 1
0052 fprintf('bayesian 90%% upper limit from weighted Statistic:\n')
0053 UL_f = r0(1,max(find(cumsum(likelihood_f/n) < confidence)))
0054
0055 %% Marginalization over calibration uncertainty
0056 % calibration range, number of points = n/10
0057 if doMargCal
0058 calib_min = 1-0.31;
0059 calib_max = 1+0.37;
0060 calib  = linspace(calib_min, calib_max, n/10);
0061 calib  = repmat(calib',1,n);
0062 y_t    = repmat(y_t,n/10,1);
0063 r0     = repmat(r0,n/10,1);
0064 sigma2 = repmat(sigma2,n/10,1);
0065 p_y_c = (1./calib) .* exp(- ((y_t .* calib).^2 + r0.^2) ./ (2*(sigma2 .* calib.^2)))
.* besseli(0,y_t.*r0./(sigma2 .* calib)); % posterior PDF
0066 p_y_c = sum(p_y_c,1);
0067 likelihood_f_c = p_y_c / sum(p_y_c/n);
```

```
0068 fprintf('bayesian 90%% upper limit from weighted Statistic
after calibration uncertainty marginalization:\n')
0069 UL_f_c = r0(1,max(find(cumsum(likelihood_f_c/n) < confidence)))
0070 end;
0071
0072 %% Marginalization over angle per Statistic
0073 r0 = linspace(0,priorMax,n); % prior PDF (flat)
0074 y  = sqrt(rstat.^2 + istat.^2); % magnitude of j-th Statistic
0075 y  = repmat(y,1,n);
0076
0077 sigma = repmat(sigma,1,n);
0078 r0    = repmat(r0,length(rstat),1);
0079
0080 posterior = exp(- (y.^2 + r0.^2) ./ (2*(sigma.^2))) .* besseli(0,y.*r0./sigma.^2);
0081
0082 w_m = mean(log(posterior),2);
0083 w   = log(posterior) - repmat(w_m,1,n);  % logarithm of posterior PDFs
0084
0085 w_t = sum(w,1);
0086 w_t = w_t - max(w_t);         % total logarithm of posterior PDF
0087
0088 likelihood_b = exp(w_t) / sum(exp(w_t)/n); % total likelihood
normalized to integrate to 1
0089
0090 fprintf('bayesian 90%% upper limit
from combined (angle-marginalized) Statistics:\n')
0091 UL_b = r0(1,max(find(cumsum(likelihood_b/n) < confidence)))
0092
0093 plot(r0(1,:),likelihood_f,'b',r0(1,:),likelihood_b,'r');grid;
0094 legend('Total likelihood after overall angle-marginalization','Total
likelihood after per-frame angle-marginalization');
0095
0096 %% Marginalization over calibration uncertainty
0097 if doMargCal
0098 r0     = linspace(0,priorMax,n);
0099 r0     = repmat(r0,n/10,1);
0100 myfun = @(x,xdata)(x(3)*exp(-((x(1)-xdata).^2)/(2*x(2)^2))); % gaussian used
for fitting
0101 xx = linspace(0,1,1000);                                    % x-axis space
(r0 prior scaled by 1/priorMax)
0102 fprintf('Marginalization over Calibration uncertainty
for combined Statistic (likelihood) \n Fit of total likelihood to a gaussian:\n')
0103 x = lsqcurvefit(myfun,[0;0.2;23],xx,likelihood_b);
0104 A  = x(3);
0105 s  = x(2)*priorMax;
0106 x0 = x(1)*priorMax;
```

```
0107 sprintf('from fit, A= %g, x0 = %g, sigma = %g',A,x0,s)
0108 A  = repmat(A,n/10,1000);
0109 x0 = repmat(x0,n/10,1000);
0110 s  = repmat(s,n/10,1000);
0111 xx = repmat(xx*priorMax,n/10,1);
0112 pos_c    = A .* exp(-(x0.*calib-xx).^2 ./ (2*(s.*calib).^2)); % posterior PDF
0113 pos_c    = (1./calib) .* pos_c;
0114 likelihood_b_c = sum(pos_c,1);   % marginalized likelihood
0115 likelihood_b_c = likelihood_b_c / sum(likelihood_b_c/n); % normalized likelihood
0116 fprintf('bayesian 90%% upper limit from combined
(angle-marginalized) Statistics after
calibration uncertainty marginalization:\n')
0117 UL_b_c = r0(1,max(find(cumsum(likelihood_b_c/n) < confidence)))
0118 end;
0119 %%
0120 results.UL_f           = UL_f;
0121 results.UL_b           = UL_b;
0122 results.likelihood_f   = likelihood_f;
0123 results.likelihood_b   = likelihood_b;
0124 results.r0             = r0(1,:);
0125 results.stats          = length(rstat);
0126 results.confidence     = confidence*100;
0127 if doMargCal
0128             results.A  = x(3);
0129             results.x0 = x(1)*priorMax;
0130             results.s  = x(2)*priorMax;
0131 results.likelihood_f_c = likelihood_f_c;
0132 results.likelihood_b_c = likelihood_b_c;
0133         results.UL_f_c = UL_f_c;
0134         results.UL_b_c = UL_b_c;
0135 end;
0136
0137 if nargout == 1,
0138     varargout{1} = results;
0139 end
0140
0141 return;
0142
0143 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
0144 likelihood_f   = results.likelihood_f;
0145 likelihood_b   = results.likelihood_b;
0146 likelihood_f_c = results.likelihood_f_c;
0147 likelihood_b_c = results.likelihood_b_c;
0148 r0             = results.r0;
0149 stats          = results.stats;
0150 UL_f           = results.UL_f;
```

```
0151 UL_b           = results.UL_b;
0152 confidence     = results.confidence;
0153 UL_f_c         = results.UL_f_c;
0154 UL_b_c         = results.UL_b_c;
0155
0156 % Plots
0157 figure;
0158 plot(r0,likelihood_f,'b',r0,likelihood_b,'r');grid;
0159 title(sprintf('%d %% UL from final weighted Statistic = %g, \n  %d %% UL
from combined Statistic = %g',confidence,UL_f,confidence,UL_b));
0160 legend(sprintf('Likelihood of final weighted Statistic \n
(after marginalization over angle)'),...
0161     sprintf('Likelihood of combined (%d Statistics) Statistic \n
(after marginalization over angle)',stats));
0162 xlim([0 0.15*10^(-44)]);
0163
0164 figure;
0165 plot(r0,likelihood_f_c,'b',r0,likelihood_b_c,'r');grid;
0166 title(sprintf('%d %% UL from final weighted Statistic = %g, \n  %d %%
UL from combined Statistic = %g',confidence,UL_f_c,confidence,UL_b_c));
0167 legend(sprintf('Likelihood of final weighted Statistic \n
(after marginalization over angle and over calibration uncertainty)'),...
0168     sprintf('Likelihood of combined (%d Statistics) Statistic \n
(after marginalization over angle and calibration uncertainty)',...
0169     stats));
0170 xlim([0 0.15*10^(-44)]);
0171
0172
0173 %           ****** Landscape2 *******
0174 % program to plot the stochastic landscape limits
0175
0176 % constants
0177 spl = 2.99792e8  ;        % speed of light in m/s
0178 gnewt = 6.6726e-11;       % Newton's constant in m^3 kg^-1 s^-2
0179 hub = 72e3/3.085678e22;   % Hubble constant in s^-1 , for 72 km/s-Mpc
0180
0181 % Input data as limits on amplitude strain density, strain Hz^-1/2
0182 f = [100 915 37520 75040];
0183 h = [2.33e-24 1.05e-23 sqrt(1.63e-46/0.113) sqrt(3.3e-48/0.0234)];
0184
0185 % Set up some graphics properties
0186
0187 set(0, ...
0188 'defaultAxesFontSize',   14, ...
0189 'defaultAxesLineWidth',   1.5, ...
0190 'defaultLineLineWidth',   0.8, ...
```

```
0191 'defaultPatchLineWidth',  0.7, ...
0192 'defaultAxesXGrid',       'on', ...
0193 'defaultAxesYGrid',       'on', ...
0194 'defaultTextFontName',    'Times', ...
0195 'defaultAxesFontName',    'Times');
0196
0197
0198 % plot strain limits
0199 figure;
0200 loglog(f,h,'bd--','Linewidth',2,'Markersize',10);
0201 axis([10 100000 1e-24 1e-22]);
0202 title('Upper limits on stochastic strain density as a funcion of frequency',
'FontSize',14);
0203 xlabel('Frequency (Hz)');
0204 ylabel('Strain/sqrt(Hz)');
0205
0206 %calculate energy density per unit frequency in J/m^3-Hz
0207 % drho/df = |h_0|^2*[(pi*f^2*c^2)/2*G_N
0208 figure;
0209 drdf = (20*pi*(spl^2)/(8*gnewt))*(h.*h).*(f.*f);
0210 loglog(f,drdf,'bd--','Linewidth',2,'Markersize',10);
0211 axis([10 100000 1e-16 1e-6])
0212 title('Upper limits on stochastic energy density as a funcion of frequency');
0213 xlabel('Frequency (Hz)');
0214 ylabel('drho/df (J/m^3-Hz)');
0215
0216 %calculate energy flux per unit frequency in Jw ( 1 Jw = 10^-6 W/m^2- Hz
0217 % dI/df = c*drdf*10^6
0218 figure;
0219 dIdf = spl*drdf*1e6;
0220 loglog(f,dIdf,'bd--','Linewidth',2,'Markersize',10);
0221 axis([10 100000 1e-2 1e10]);
10);
0211 axis([10 100000 1e-16 1e-6])
0212 title('Upper limits on stochastic energy density as a funcion of frequency');
0213 xlabel('Frequency (Hz)');
0214 ylabel('drho/df (J/m^3-Hz)');
0215
0216 %calculate energy flux per unit frequency in Jw ( 1 Jw = 10^-6 W/m^2- Hz
0217 % dI/df = c*drdf*10^6
0218 figure;
0219 dIdf = spl*drdf*1e6;
0220 loglog(f,dIdf,'bd--','Linewidth',2,'Markersize',10);
0221 axis([10 100000 1e-2 1e10]);
0222 title('Upper limits on stochastic energy flux as a funcion of frequency');
0223 xlabel('Frequency (Hz)');
```

```
0224 ylabel('dI/df (Jw = 10^-6 W/m^3-Hz)');
0225
0226 %calculate Omega
0227 % Omega = (20*pi^2/(3*hub^2))*h^2*f^3
0228 figure;
0229 Omega = (20*(pi^2)/(3*(hub^2)))*(h.*h).*(f.*f.*f);
0230 loglog(f,Omega,'bd--','Linewidth',2,'Markersize',10);
0231 axis([10 100000 1e-5 1e8]);
0232 title('Upper limits on Omega as a funcion of frequency- h_0 = 72');
0233 xlabel('Frequency (Hz)');
0234 ylabel('Omega');
0235
0236 answer = f
0237 answer = h
0238 answer = drdf
0239 answer = dIdf
0240 answer = Omega
0241
0242
```