A “Multiple-interferometer pipelines” document

Team members
Sukanta Bose, Patrick Brady, Nelson Christensen, Jolien Creighton, Teviet Creighton, Harald Lück, Benoîr Mours, David Reitze

Flowchart of a code for evaluating the coherence statistic:

LALInspiralCoherenceSearch(): Main Program or Wrapper

If analysis involves a single interferometer, then run the single interferometer inspiral search code LALFindChirp(),
else
if analysis involves two (non-identically oriented) interferometers, then run
LALInspiralTwoIntfCoherenceSearch(),
else
if analysis involves three or more (arbitrarily oriented) interferometers, then run
LALInspiralMultiIntfCoherenceSearch().

LALInspiralTwoIntfCoherenceSearch()

This code takes two GW channel data streams, each with n sample points, from two non-aligned, non-coincident interferometers (call them 1 and 2) and computes the coherence statistic as a function of time:

\[ L_{12}(t_i; \tau_j) = \left( |C_1(t_i)|^2 + |C_2(t_i; \tau_j)|^2 \right)^{1/2} \]

\[ = \left( |X_1(t_i)|^2 + |Y_1(t_i)|^2 + |X_2(t_i; \tau_j)|^2 + |Y_2(t_i; \tau_j)|^2 \right)^{1/2} \]

where \( i = 0, \ldots, (n-1) \) and \( j = 0, \ldots, (m-1) \), such that \( \tau_{(m-1)} \geq \text{light travel time between the two interferometers} \). Also, \( \tau_j - \tau_{(j-1)} \equiv \Delta \tau^2 \), which is independent of \( j \); it is determined by the noise PSDs of the two interferometers and the minimal match between neighboring network templates in the parameter space comprising of the two masses, \( \{m_1, m_2\} \) and the source-direction angle, \( \theta \).

Note that the antenna-pattern functions of the interferometers are not needed to compute this statistic. In fact \( L_{12}(t_i; \tau_j) \) can be computed purely from (i) the site location information of the two interferometers and (ii) the outputs \( \{X(t_i), Y(t_i), \chi^2(t_i)\} \) of the single detector search code (LALFindChirp()) filtering two data streams with a single template. For every \( i \) and \( j \), \( L_{12}(t_i; \tau_j) \) is compared with a preset threshold \( L^1_{12} \). Similarly, the chi-square veto is implemented at the level of an individual detector by comparing \( \chi^2(t_i) \) with the preset threshold \( \chi^2_0 \), for every \( i \).

\(^1\)Caution: This document is still under preparation. Please watch out for bugs! Dated: April 19, 2001.

\(^2\)See, e.g., the document on coherence statistic posted at http://www.aei.mpg.de/~bose.
Method 1

1. while \((i <= m - 2)\) {

   /* run LALFindChirp() on data from detector 2 */

   compute \(\{X_2(t_i), Y_2(t_i), \chi_2^2(t_i)\}\);

   Store these in an array;

}

2. while \((i <= n)\) {

   /* run LALFindChirp() on data from detector 1 */

   compute \(\{X_1(t_i), Y_1(t_i), \chi_1^2(t_i)\}\);

   /* run LALFindChirp() on data from detector 2 */

   compute \(\{X_2(t_i; \tau_{m-1+i}), Y_2(t_i; \tau_{m-1+i}), \chi_2^2(t_i; \tau_{m-1+i})\}\);

   /* select coincidence window */

   for \((j = (i < m) \ ? \ (i - m); \ j <= i + m - 1; \ j++)\) {

      /* implement chi-square veto test */

      if \((\chi_1^2(t_i) < \chi_{10}^2 \text{ and } \chi_2^2(t_j) < \chi_{20}^2)\) {

         /* compute coherence statistic */

         if \((I_{12}(t_i; \tau_j) > I_{12}^0)\)
               store event/filter info;
         else continue;

      } else continue;

   }

}
In the above codes, \( \text{LAL} \) is to be interpreted as a LAL function that computes Eq. (1) from the outputs of \text{LALFindChirp}() acting on the data from two detectors. Also, “event/filter info” comprises \( \{ X_i(t_i), Y_i(t_i), \chi^2_i(t_i) \} \), \( \{ X_2(t_i; \tau_m), Y_2(t_i; \tau_m), \chi^2_2(t_i; \tau_m) \} \), and the parameters of the template being used by \text{LALFindChirp}().

\( \text{LALInspiralMultiIntfCoherenceSearch}() \)

This code takes \( M \) number of GW channel data streams, each with \( n \) sample points, from arbitrarily oriented (non-coincident) interferometers and computes the coherence statistic as a function of time:

\[
M(t_i) = \left( |C_+(t_i)|^2 + |C_-(t_i)|^2 \right)^{1/2}
= \left( |X_+(t_i)|^2 + |Y_+(t_i)|^2 + |X_-(t_i)|^2 + |Y_-(t_i)|^2 \right)^{1/2},
\]

where \( i = 0, \ldots, (n - 1) \) and

\[
X_{\pm}(t_i) := \sum_{I=1}^{M} X_I v^I_{\pm},
Y_{\pm}(t_i) := \sum_{I=1}^{M} Y_I v^I_{\pm}.
\]

Here \( v^I_{\pm} \) are components of the two orthonormal “helicity” basis vectors, \( \hat{v}_{\pm} \).

\( \text{LALInspiralHelicity}() \)

The \( \hat{v}_{\pm} \) are determined by the source direction, \( \{ \theta, \phi \} \), and the detector orientations. Therefore, the temporal evolution of the antenna-pattern functions of the interferometers is necessary to compute the above statistic. The (tentatively termed) function \text{LALInspiralHelicity}() is being coded to input \( \{ \theta, \phi \} \) and the temporal evolution of the antenna-pattern functions for a given network from, say, David Chin’s code (assuming it will be in LAL soon) to compute \( \hat{v}_{\pm} \).

\( \text{LALInspiralTimeLag}() \)

Given site location information of \( M \) interferometers, this code will compute the light travel time between site \( I \) and Earth’s center, for a given source direction, \( \{ \theta, \phi \} \):

\[
\tau_{(I)}(\theta, \phi) = \frac{(r_{(I)} - r_0) \cdot \hat{n}(\theta, \phi)}{c},
\]

where \( r_{(I)} \) and \( r_0 \) are the position vectors of the \( I \)-th detector and Earth’s center, respectively, in any given reference frame; \( c \) is the speed of light. Note that \( \tau_{(I)}(\theta, \phi) \) can take positive as well as negative values. Does such a code already exist in LAL?
LALInspiralSourceDirecGrid()

This code creates a two-dimensional lattice on the parameter space \{\theta, \phi\}. The spacing between neighboring lattice points is determined by the noise PSDs of the interferometers and the minimal match chosen for neighboring templates.

Sample (piece of) LALInspiralMultiIntCoherenceSearch()

while (i <= n) {
    for (I = 1; I <= M; I++) {
        /* run LALFindChirp() on data from detector 1 to M */
        compute \{X_I(t_i), Y_I(t_i), \chi^2_I(t_i)\};
    }

    while (k <= maximum available value on \{\theta, \phi\} grid) {
        choose a point, \{\theta_k, \phi_k\}, in the \{\theta, \phi\} grid;
        /* run LALInspiralHelicity() for \{\theta_k, \phi_k\} and t_i */
        compute \hat{\phi}_\pm(\theta_k, \phi_k; t_i);
        /* run LALInspiralTimeLag() for \{\theta_k, \phi_k\} */
        compute \tau_{(I)}(\theta_k, \phi_k);

        /* implement chi-square veto test */
        if (\chi^2_I(t_i; \tau_{(I)}) < \chi^2_{0}, for all I) {
            /* compute coherence statistic */
            if (L_M(t_i; \theta_k, \phi_k) > I^0_{MT})
                store event/filter info;
            else continue;
        }
        else continue;
    }
}
In the above code, the chi-square veto is implemented at the level of an individual detector by comparing \( \chi^2(t_i; \tau_{ij}) \) with the preset threshold \( \chi^2_{\text{Threshold}} \). The function “\( L_M(t_i; \theta_k, \phi_k) \)” is to be interpreted as a LAL function that computes Eq. (2) from the outputs of \texttt{LAL\textunderscore Find\textunderscore Chirp()} acting on the data from \( M \) detectors. Its value is compared with a preset threshold \( L_M^0 \). Also, “event/filter info” comprises of \( \{X_I(t_i), Y_I(t_i), \chi^2_I(t_i)\} \), for all \( I \), and the parameters of the template used by \texttt{LAL\textunderscore Find\textunderscore Chirp()}. 