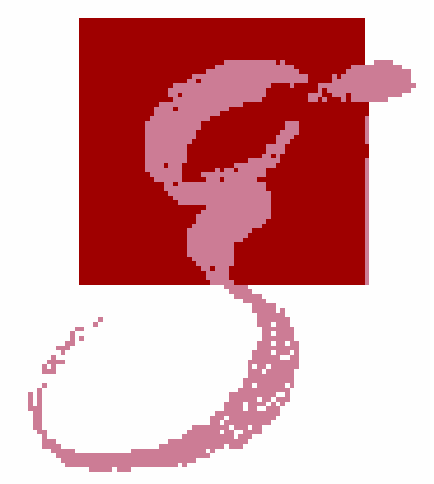
The Hough Transform Algorithm

–code status report–



Alicia M. Sintes

Max-Planck-Institut für Gravitationsphysik
Albert-Einstein-Institut
Golm, Germany
http://www.aei-potsdam.mpg.de

LSC meeting. August 13-16, 2001

LIGO-G010341-00-Z

The Hough Transform

The Hough transform is a transformation between the data and the space of parameters that describe the signal.

Input: Set of points in time-frequency plane
(from DeFT according to a sky location & spin-down)
- A source located at the center of the patch with the same spin-down parameters will appear as a set of points forming an horizontal line at the intrinsic frequency of the source.
- Due to the mismatch, the points will appear in patterns following the so called: *Hough transform master equation*.

Output: Histograms in the parameter space:
for each f_0, residual spin-down and refined sky location.

For every point in the t-f plane, one enhances the number count in the histogram in the pixels that are consistent.

Significant clustering in a pixel in parameter space indicates likelihood of the data containing a signal having the parameters of that pixel.

Master equation & implementation

$$\boldsymbol{\nu - F_0 = \vec{\xi}(t) \cdot (\hat{n} - \hat{N})}$$

$$F_0 \equiv f_0 + \sum_k \Delta f_k \left[\Delta T\right]^k \ , \quad \Delta T \equiv T_{\hat{N}}(t) - T_{\hat{N}}(\hat{t}_0).$$

ν: measured frequency of the signal at time t
f_0: intrinsic frequency of the signal
$T_{\hat{N}}(t)$: time at the SSB for a given sky location $\hat{N}$
$\Delta f_k = f_k - F_k$: residual sin-down parameter.

$$\vec{\xi}(t) = \left(F_0 + \sum_k F_k \left[\Delta T\right]^k \right) \frac{\vec{v}(t)}{c} + \left(\sum_k k F_k \left[\Delta T\right]^{k-1} \right) \frac{\vec{x}(t) - \vec{x}(\hat{t}_0)}{c}$$

Implementation issues:

- The *Hough map (HM)* is an histogram, thus additive. It can be seen as the sum of several partial maps *(PHM)* constructed using the *peaks* from just one periodogram.
- The effect of the *residual spin-down* parameter is just a change in F_0. At any given time, F_0 can be considered constant.
- The *HM* corresponding to $\{f_0, f_k\}$ is constructed by adding, at different times, *PHM* with different F_0 values.

Milestones report Feb-2001

Incoherent stage:

we have benchmarked the different moduli of the *Hough code* (presented at the August'00 LSC meeting) based on the *zooming algorithm* on different platforms and compilers, and we have identified the bottleneck that prevents us from obtaining the desired performance in particular cases.

We have investigated the feasibility of improvement by considering alternative methods. The study suggested the development of a *new algorithm*, based on **stereographic projection**, and the construction of **partial look up tables (LUT)** - partial in the sense that they are valid only for a given sky patch, time-stamp and frequency range.

We expect a significant improvement in performance from this new implementation in the case of non-residual spin-down.

The implication of such an alternative is that we should change the core of the Hough code, and also the different moduli, accordingly. The most tricky part is the construction of the correct LUT in which portions of annuli are already clipped on the sky patch under study, thus speeding up the subsequent process of construction of a PHM. The natural sky tiling is somewhat different from the one used by the zooming algorithm thus also this part needs to be changed.

Milestones Feb - Aug 2001

– Incoherent stage –

a. Construction of a partial-LUT (Look-Up-Table)

b. Construction of PHM, represented as a list of pointers to star-stop tables for the different annuli.

c. Update the representation of the CONE

d. Construction of a HM-derivative

d. Integration of a HM-derivative to build a HM

f. Development of the circular-buffer for updating the CONE

g. Update of the LUT when changing frequency range (DrC)

h. Development of a separate code for the case of non-residual spin-down. $(integrated)$

i. Benchmark the new code modulus.

j. Routines to perform rotations on the sphere: the coordinate change necessary to move the center of the sky patch under study to the origin of the Cartesian reference frame on the projected plane (and the inverse).

k. Routine to convert the demodulation parameters in the input parameters necessary for LUT construction.

l. Stereographic projection routines: from sphere to plane and inverse.

m. Optionally: study the feasibility of 90 degrees rotations of the LUT for different times in a year. (no)

Code status

The package is organized under the following headers

	Modules
LUT.h	Stereographic.c PatchGrid.c ParamPLUT.c ConstructPLUT.c TestConstructPLUT.c
PHMD.h	Peak2PHMD.c TestPeak2PHMD.c
HoughMap.h	HoughMap.c TestHoughMap.c
LALHough.h	DriveHough.c TestDriveHough.c

The package is written following the LAL specifications.
It was compiled with lal-0.8.
Documentation is missing.

Prototypes

- Header LUT.h

Tiling the sky patch:

```
void LALHOUGHPatchGrid (
        LALStatus            *status,
        HOUGHPatchGrid       *out,  /* Patch Grid info */
        HOUGHResolutionPar   *in1
        );
```

Conversion of demodulation parameters in the input parameters necessary for a partial-LUT:

```
void LALHOUGHParamPLUT (
        LALStatus         *status,
        HOUGHParamPLUT    *out,  /* parameters needed build LUT*/
        INT8              f0Bin, /* freq. bin to construct LUT */
        HOUGHDemodPar     *par   /* demodulation parameters */
        );
```

Construction of a partial-LUT:

```
void LALHOUGHConstructPLUT (
        LALStatus        *status,
        HOUGHptfLUT      *lut,
        HOUGHPatchGrid   *patch,
        HOUGHParamPLUT   *par
        );
```

Stereographic projection routines:
from the sphere to the plane and inverse

```
void LALStereoProjectPolar (
        LALStatus           *status,
        REAL8Polar2Coor     *out,
        REAL8UnitPolarCoor  *in
        );

void LALStereoProjectCart (
        LALStatus           *status,
        REAL8Cart2Coor      *out,
        REAL8UnitPolarCoor  *in
        );

void LALStereoInvProjectPolar (
        LALStatus           *status,
        REAL8UnitPolarCoor  *out,
        REAL8Polar2Coor     *in
        );

void LALStereoInvProjectCart (
        LALStatus           *status,
        REAL8UnitPolarCoor  *out,
        REAL8Cart2Coor      *in
        );
```

Sphere rotations:
Coordinate changes to move the center of the sky patch to the origin of the Cartesian reference frame on the projected plane (and the inverse)

```
void LALRotatePolarU (
        LALStatus             *status,
        REAL8UnitPolarCoor    *out,
        REAL8UnitPolarCoor    *in,
        REAL8UnitPolarCoor    *par
        );

void LALInvRotatePolarU (
        LALStatus             *status,
        REAL8UnitPolarCoor    *out,
        REAL8UnitPolarCoor    *in,
        REAL8UnitPolarCoor    *par
        );
```

- Header `PHMD.h`

Construction of a partial map (PHMD)
as a list of pointers to the borders of the annuli in the LUT, using the peaks from one periodogram

```
void LALHOUGHPeak2PHMD (
        LALStatus       *status,
        HOUGHphmd       *phmd,
        HOUGHptfLUT     *lut,
        HOUGHPeakGram   *pg
        );
```

- Header HoughMap.h

Initialization of a HM-derivative:

```
void LALHOUGHInitializeHD (
        LALStatus       *status,
        HOUGHMapDeriv   *hd /* the Hough map derivative */
        );
```

Construction of a HD as a sum of several PHMD:

```
void LALHOUGHAddPHMD2HD (
        LALStatus      *status,
        HOUGHMapDeriv  *hd,  /* the Hough map derivative */
        HOUGHphmd      *phmd  /* info from a partial map */
        );
```

Initialization of the Hough map:

```
void LALHOUGHInitializeHT (
        LALStatus       *status,
        HOUGHMapTotal   *ht,     /* the total Hough map */
        HOUGHPatchGrid  *patch     /* patch information */
        );
```

Integration of a HM-derivative to build a Hough map:

```
void LALHOUGHIntegrHD2HT (
        LALStatus       *status,
        HOUGHMapTotal   *ht,     /* the total Hough map */
        HOUGHMapDeriv   *hd /* the Hough map derivative */
        );
```

- Header LALHough.h

Construction of the space of PHMD represented by a cylinder (old CONE)

```
void LALHOUGHConstructSpacePHMD (
        LALStatus             *status,
        PHMDVectorSequence    *phmdVS,
        HOUGHPeakGramVector   *pgV,
        HOUGHptfLUTVector     *lutV
        );
```

Development of a circular buffer for updating the cylinder as frequency changes:

```
void LALHOUGHupdateSpacePHMDup (
        LALStatus             *status,
        PHMDVectorSequence    *phmdVS,
        HOUGHPeakGramVector   *pgV,
        HOUGHptfLUTVector     *lutV
        );

void LALHOUGHupdateSpacePHMDdn (
        LALStatus             *status,
        PHMDVectorSequence    *phmdVS,
        HOUGHPeakGramVector   *pgV,
        HOUGHptfLUTVector     *lutV
        );
```

Construction of a total Hough map from the PHMD-cylinder and a list of frequency indexes

```
void LALHOUGHConstructHMT (
        LALStatus                   *status,
        HOUGHMapTotal               *ht,
        UINT8FrequencyIndexVector   *freqInd,
        PHMDVectorSequence          *phmdVS
        );
```

Computation of the frequency index of the PHMD, given f_0 and possible residual spin parameters (valid for any residual-spin order)

```
void LALHOUGHComputeFBinMap (
        LALStatus              *status,
        UINT8                  *fBinMap,
        UINT8                  *f0Bin,
        HOUGHResidualSpinPar   *rs
        );
```

Hough transform:

– illustration –