The ***STACK-SLIDE*** Search

Update: LSC August 2003

Gregory Mendell, Mike Landry

LIGO Hanford Observatory

LIGO-G030397-00-W

Brady/Creighton algorithm with modifications

First version of code will stackslide power from SFTs; code will ultimately stackslide the F-statistic from LALDemod.

The algorithm shown in the Flowchart is iterated in a hierarchical approach.

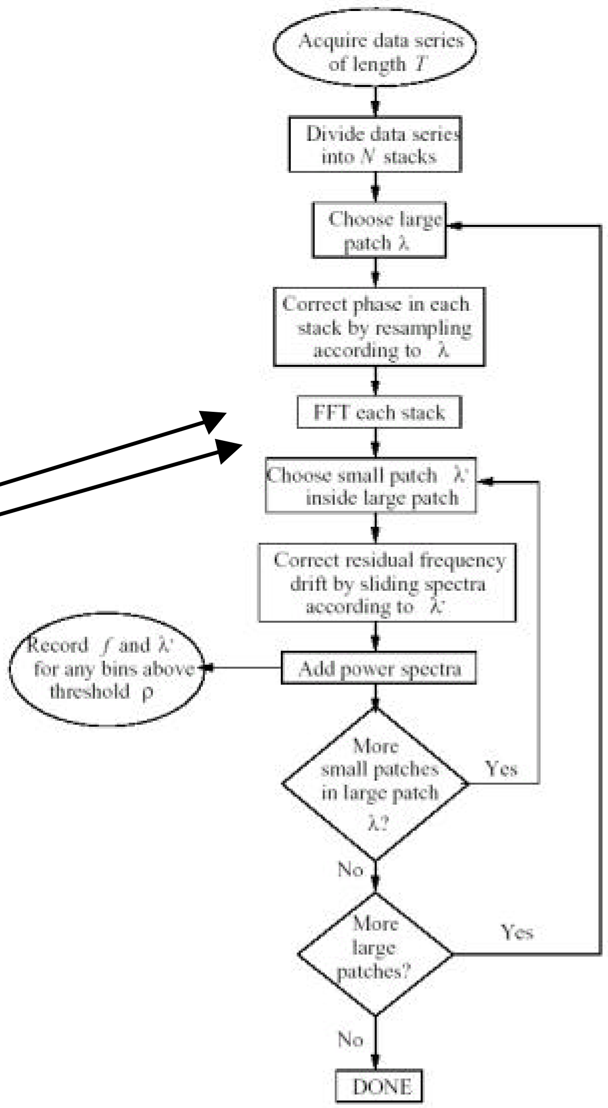


FIG. 1.   A flowchart representation of the *stacked slide* algorithm to search for sources of continuous gravitational waves. Notice that the computational cost of sampling the fine grid is reduced by sliding the power spectra, rather than re-computing an FFT for each point on the fine grid.

Brady/Creighton gr-qc/9812014

We are writing LALWrapper, LAL, & Driver Code

- LALWrapper part will run under LDAS as DSO or under stand-alone wrapper.
- Code will search parameter space and put candidates into database.
- LAL parameter-space metric code exists.
- Need to write LAL STACK-SLIDE function.
- Driver will be user-friendly code that anyone can run.

LIGO

STACK-SLIDE Search

$$x = s + n = A\cos(2pft + f_0) + n$$

$$\tilde{x} = DFT(x)$$

One stack:
(N samples)

$$\tilde{x}^*\tilde{x} = \left(\frac{AN}{2}\right)^2 + \left(Ns^2 \pm Ns^2\right) + cross - term$$

(Ave M stacks)

$$\langle \tilde{n}^*\tilde{n} \rangle = Ns^2, \sqrt{\mathrm{var}(\tilde{n}^*\tilde{n})} = Ns^2/\sqrt{M}$$

For A²N < s² :

$$\sqrt{\frac{\langle \tilde{x}^*\tilde{x} \rangle}{\langle \tilde{n}^*\tilde{n} \rangle}} \sim (1 \pm 0.5/\sqrt{M}) + \frac{1}{8}\frac{A^2 N}{s^2}$$

Detection if:

$$\frac{A^2}{4s^2} NM^{\frac{1}{2}} >> 1 \Rightarrow \frac{A}{\sqrt{4s}} \sqrt{N} M^{\frac{1}{4}} >> 1$$

False Alarm Rate:

$$C_{false} = 1 - \frac{g\left(M, \sum P_n / S_n\right)}{(M-1)!}$$

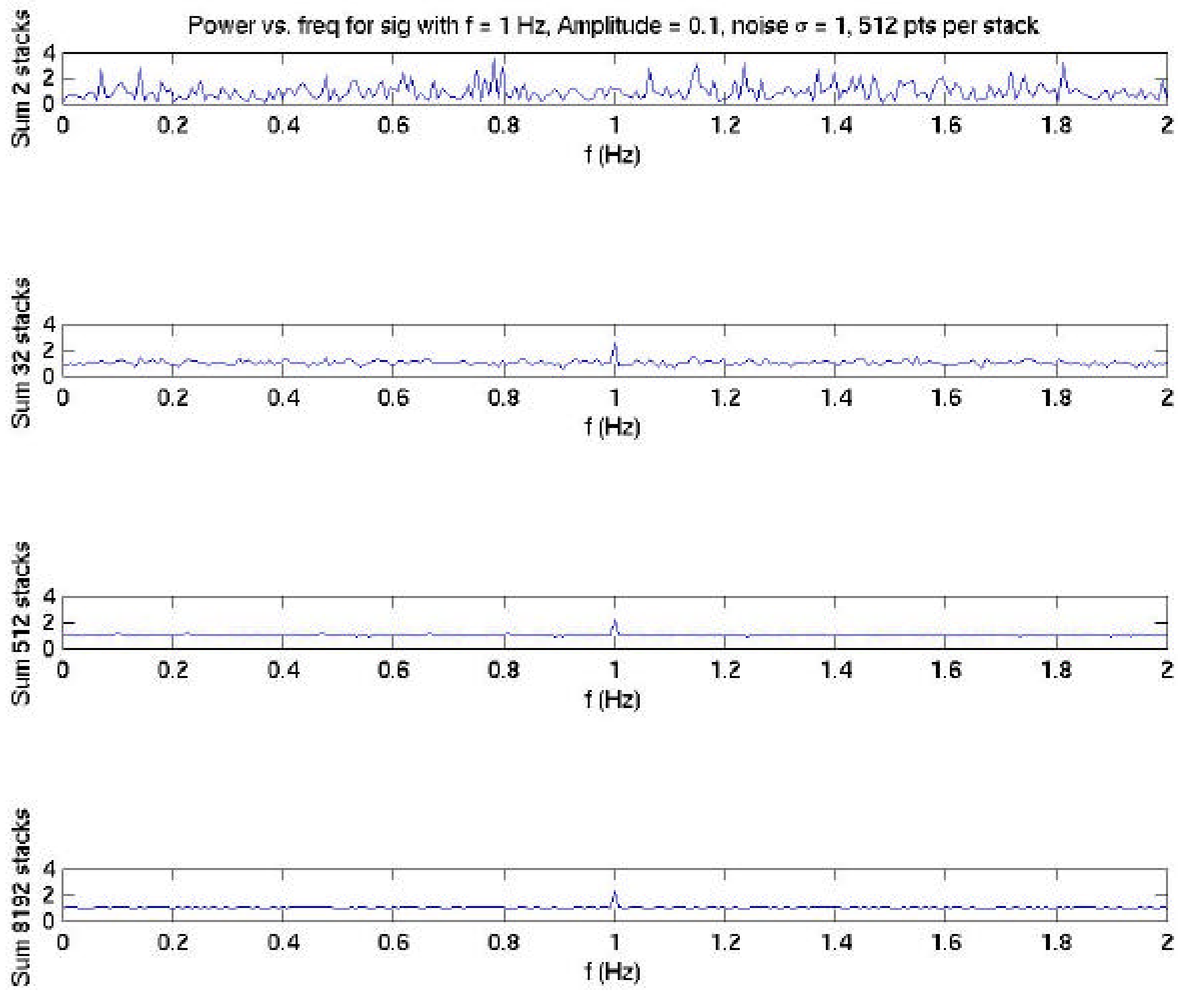
¡ = Incomplete Gamma Function

← *Brady/Creighton gr-qc/9812014*

If F-statistic: ↓

$$\propto g\left(2M, \sum F_n\right)$$

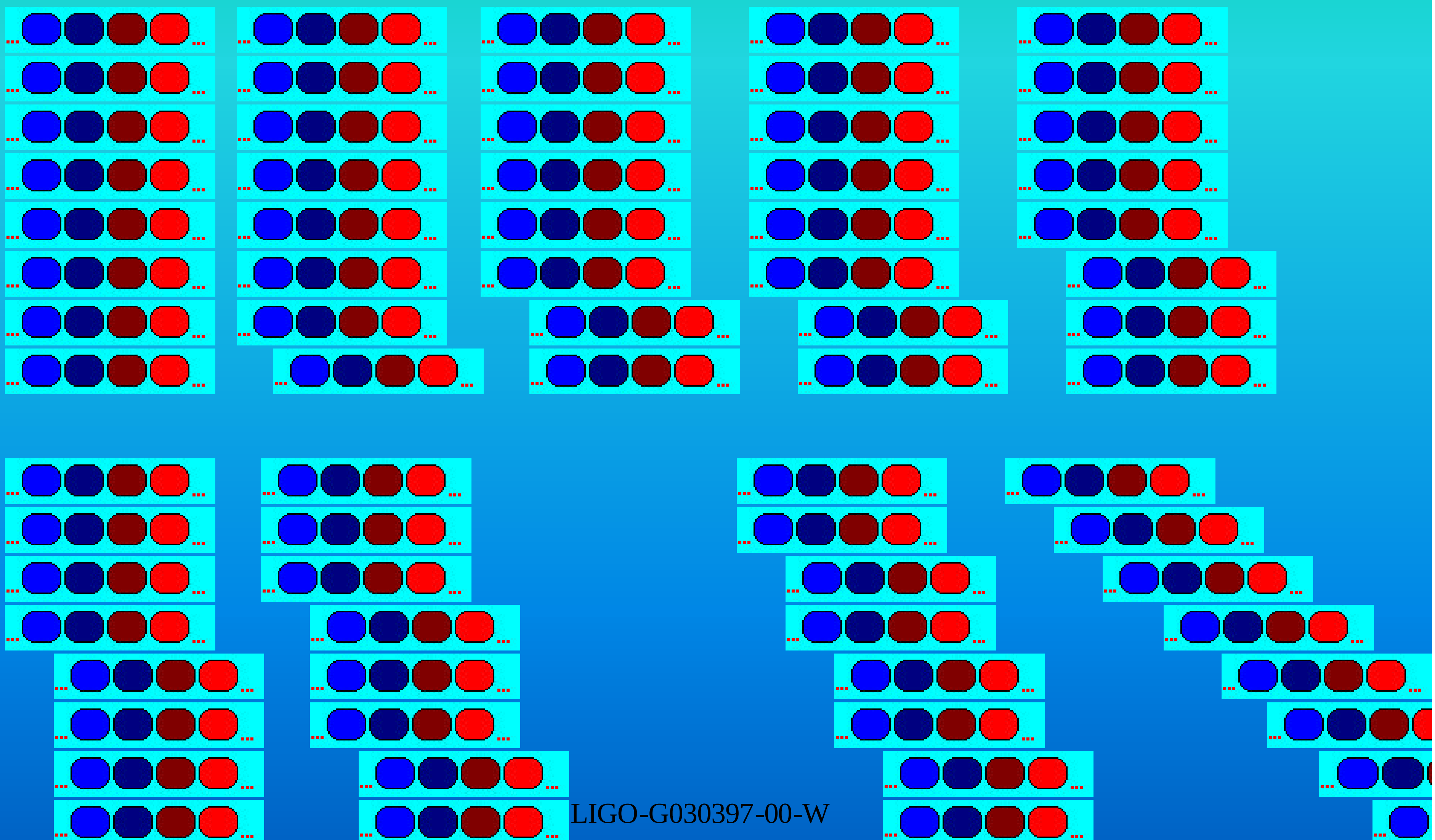
STACK-SLIDE Power vs. Freq



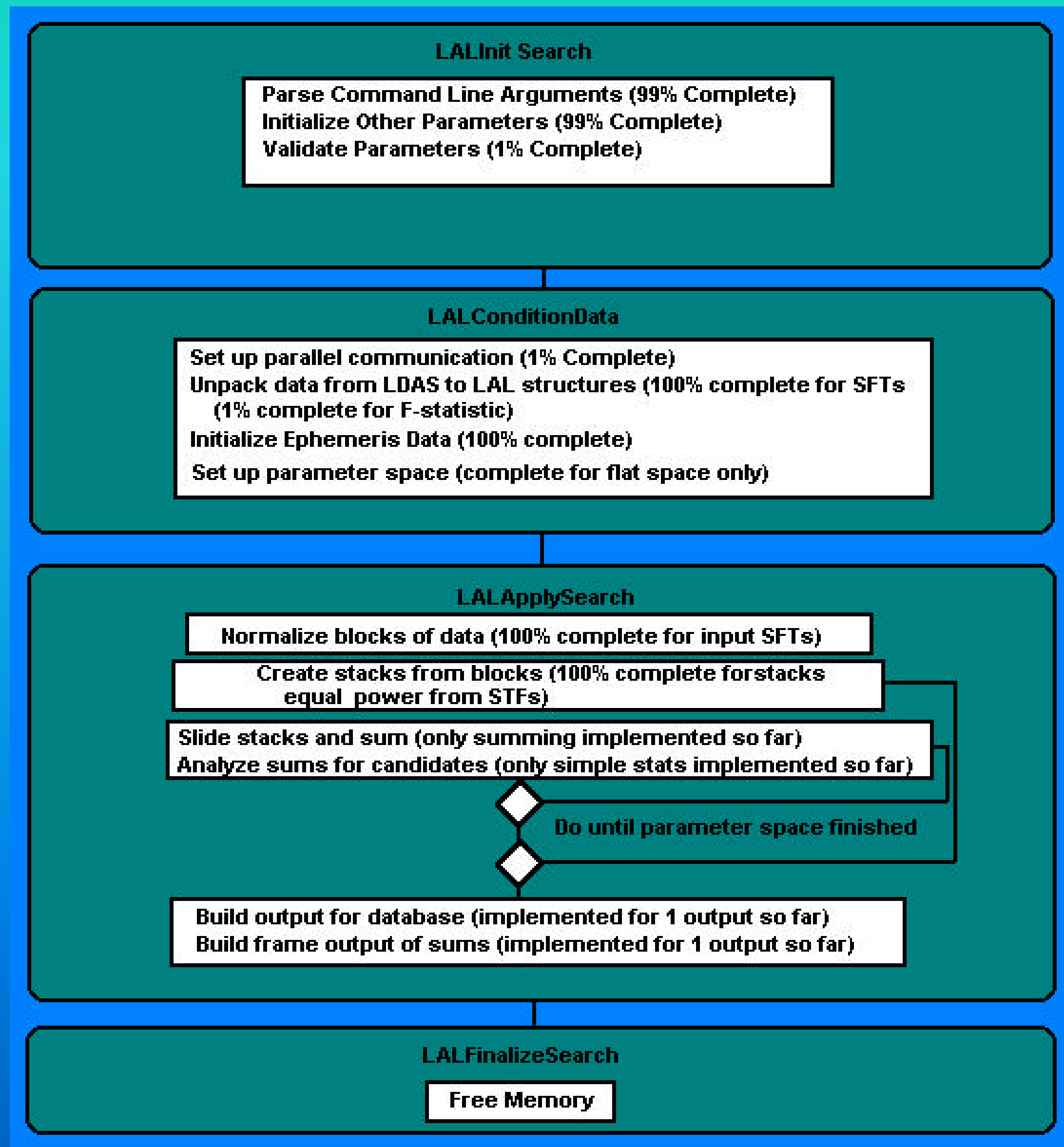
Rough Outline

- Input blocks of data.
- Coherently compute stacks from blocks for each coarse parameter space patch.
- Slide stacks and sum for each fine patch within each coarse patch.
- Statistically test sums for candidates.
- Output results to database and frame files.

Stacking and Sliding…



Progress to Date



Will be iterating…

- Writing of code (still lots to write).
- Testing and debugging.
- Refining algorithm.
- Studying computational complexity.
- Studying statistics (how to analyze results).