A versatile code interface for (merger) waveforms

Frank Ohme
(for the LVK Interface Team)

May 7 2024

Contributors

(Chinmay Kalaghatgi)

Cecilio Garcia-Quiros

Jolien Creighton

Jonathan Thompson

(Tomas Andrade)

(Stefano Schmidt)

(José Francisco Nuño)

(Sergei Ossokine)

(Angela Borchers)

Frank Ohme

Krishnendu NV

Max Melching

We need more people to engage regularly with the interface development efforts.

Continuous, even if small, time commitments are needed.

Why code interface?

- Starting point: computer code exists for waveform model
- How do we use it?

```
Wf_data = FranksModel(par1, par2, par3, par4, ...)
```

Why code interface?

Directly using others' code is time consuming. With every model, one would need to adapt to

1. Code language, function name
2. Output structure
3. Calling patterns, input parameters

-> Not scalable to many models. Time consuming to adapt to modifications.

LALSimulation - previous interface

Up until the last LVK observing run, the following interface was used as part of LALSimulation.

```
hplus, hcross = XLALSimInspiralChooseTDWaveform(m1, m2, S1x, S1y, S1z,
S2x, S2y, S2z, distance, inclination, phiRef, longAscNodes,
eccentricity, meanPerAno, deltaT, f_min, f_ref, *params, approximant)
```

- Standardized output, required parameters, optional parameters, approximant identified as string

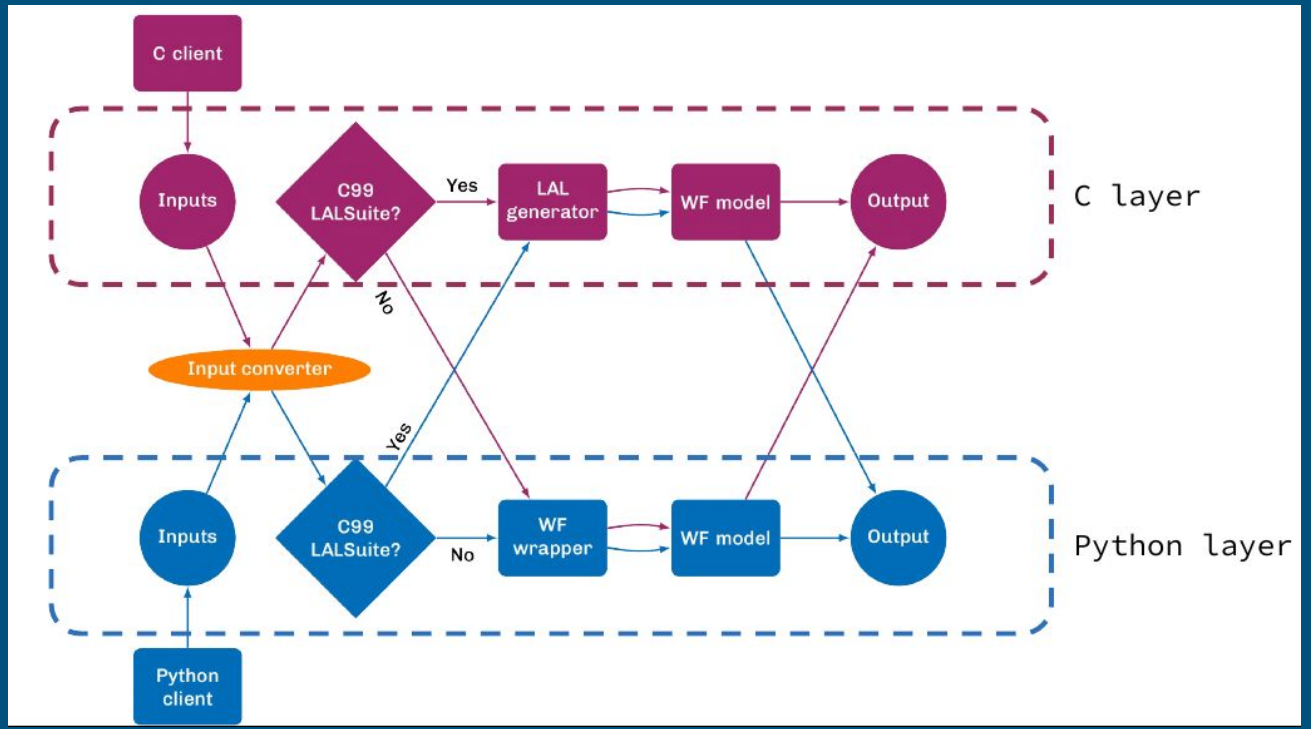
Motivation for a New Interface

Why was this not good enough?

- Static input/output structure has its limitations. Desire to add more parameters, become more user friendly, return other/more data
- Only models in LALSimulation C code were supported. Desire to become more flexible, first by integrating python code in addition to swig
- General overhaul of code base and documentation. Support reviews, modularity.

Design document

Report created by Interface committee, chaired by Serguei Ossokine



New Interface Structure

```
gwpol = GenerateTDWaveform(wfparams, generator)
```

- wfparams: flexible dictionary of input parameters (specifying the source and signal properties)
- generator: waveform model generator. Must be instantiated once with all information needed to specify the exact model. May carry auxiliary data.
- Documentation
- Returns special polarizations class with holds h_+, h_x as GWpy series

Input parameters

- Python dictionary
- Astropy units
- Mixture of standard parameter names and additional, model-specific parameters
- Some parameters may be specified in different ways, e.g.,
  - Masses (individual, chirp mass, mass ratio, mass difference)
  - Spins (Cartesian, spherical)
- Defaults introduced for certain parameters - this can be non-trivial, e.g., for spins

Interface functions

- The following should be supported in the short term
  - Time domain polarizations
  - Frequency domain polarizations
  - Time domain spherical harmonic modes
  - Frequency domain spherical harmonic modes

- More could become available in the future
  - Internal dynamics
  - Whitened signals
  - Advanced source and signal information (time-domain length of frequency-domain models, merger frequency, remnant properties, …)
  - Signal uncertainties
  - Modes in different frames (co-rotating, etc.)
  - Selectable output formats

Adding new models

- Adding new models means providing a new generator that respects a basic structure
- Approximant identified by module/package name and Class (and additional options for instantiation, if applicable)
- Information about collaboration reviewed models needs to be accessible and verifiable
- Set of standard parameter names may be extended over time

Summary

- New versatile interface supporting C and python implemented as part of LALSimulation
- More features are being added right now
- New release (~within 6 months) contains first version of fully reviewed python infrastructure `gwsignal` (`documentation`), including advanced parameter handling, conditioning routines
- Designed to work with external (python) waveform packages and LALSimulation
- Installable through LALSuite (including via `conda` and `pip`)

Publication

- Target journal: Journal of Open Source Software
- Very brief summary of code features and needs
- Coming when next release is ready