

Programming Language Interoperability

CCQ-hosted Sciware


Nils Wentzell, Matthew Fishman, Lehman Garrison, Olivier Parcollet

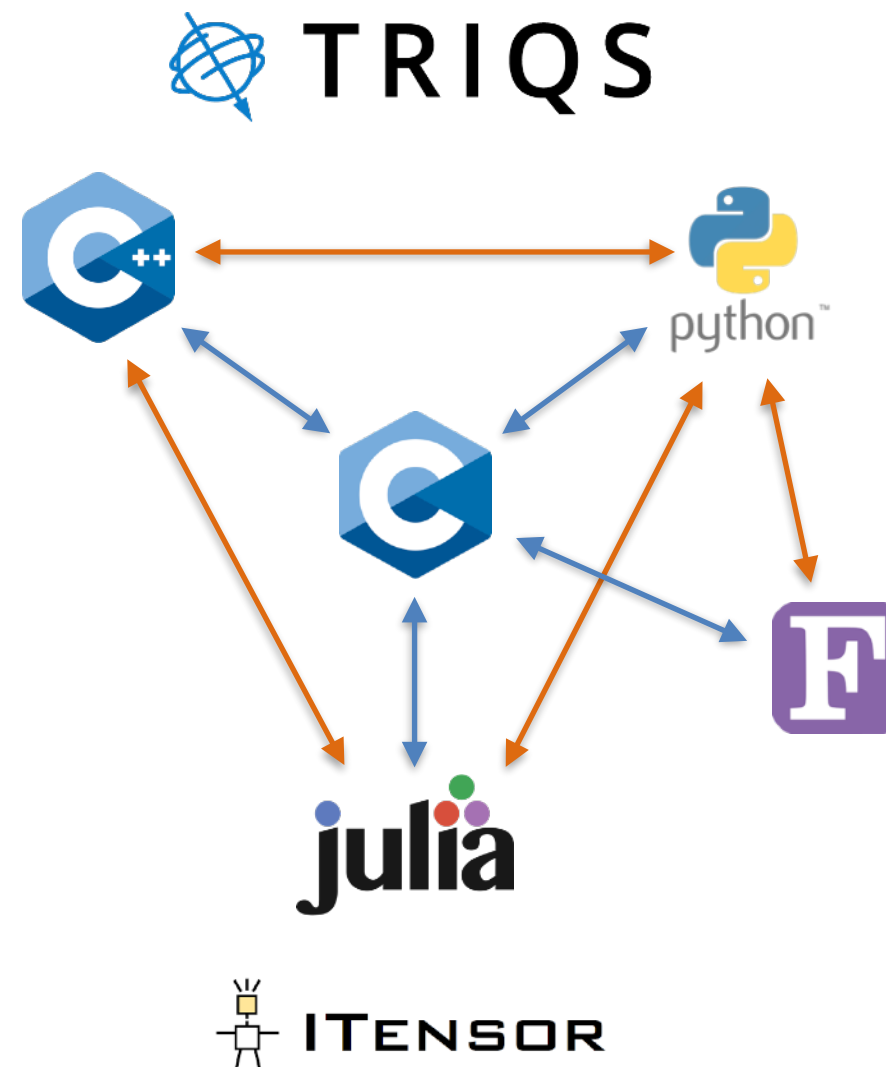
Introduction











- Programming Language Interoperation

```
import numpy as np
from ... import norm
v = np.random.rand(1e6)
print(norm(v))
```

```
double norm(span<double> v){
    // a costly computation...
    return res;
}
```

- Type Conversion, Object Lifetime, Wrapped Types
- Why interoperate between Languages?
 - Performance Improvements
 - Use of existing libraries
- Native connection through 
- How to avoid indirection?

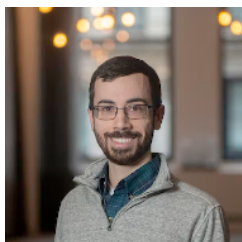


from to					
		C-Interfaces	ISO_C_BINDING	nanobind / pybind11 clair Python C-API	ccall CxxWrap.jl
	Native			ctypes Python C-API	ccall Clang.jl
	ISO_C_BINDING			f2py	ccall
	Python C-API				PythonCall
	jluna PackageCompiler.jl	PackageCompiler.jl		JuliaCall	

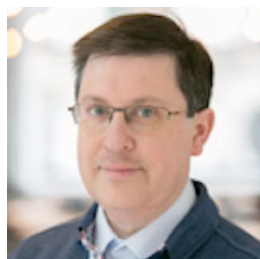
Overview



- Matthew Fishman, Research Scientist, CCQ
- PythonCall, JuliaCall, ccall, Clang.jl, PackageCompiler.jl



- Lehman Garrison, Software Engineer, SCC
- nanobind, ctypes



- Olivier Parcollet, Senior Research Scientist, CCQ
- clair

