I originally wrote this library for the genSTARK project, but it evolved into a pretty nifty stand-alone module. The library is here:

GitHub

GuildOfWeavers/galois

Arithmetic and polynomial operations in finite fields. - GuildOfWeavers/galois

High-level features are:

- Basic modular arithmetic (addition, subtraction, multiplication, inversion, exponentiation).
- Bulk operations on vectors and matrixes.
- Basic polynomial operations (addition, subtraction, multiplication, division).
- Polynomial evaluation and interpolation (using Lagrange and FFT methods).

At this point, only prime fields are supported.

WebAssembly optimization

One of the cool features of the library is a flexible optimization architecture. It is pretty simple to write optimization modules for different types of fields.

So far, I wrote an optimization module in WASM for 128-bit prime fields with modulus of the form 2^{128}-k

, where k < 2^{64}

.This resulted in the overall speed of up to 6x - 10x

as compared to native JavaScript implementation. Here are some high-level benchmarks run on Intel Core i5-7300U @ 2.60GHz (single thread):

Operations/sec

JS BigInt (256-bit)

JS BigInt (128-bit)

WASM (128-bit)

Additions

3,200,000

5,000,000

44,000,000

Multiplications

950,000

1,850,000

16,300,000

Exponentiations

3,200

10,500

97,000

WASM performance can be optimized further. Specifically, SIMD and multi-threaded evaluation is something that I’m planning to implement at some point in the future (once support for these in WASM becomes more mature). But even as is, I believe the numbers are within 2x - 4x of what can be achieved with a native C implementation.

The library is still very new, and there are a bunch of things to fix and improve (see the issues in the repo). So, would appreciate any feedback, help, and support.