

A vibrant sunset over a city skyline, featuring a mix of modern skyscrapers and older architectural styles. In the foreground, a large teal-colored bridge spans a body of water. The sky is filled with warm orange, yellow, and pink hues.

node INTERACTIVE

The word "node" is composed of four white icons: a house-like shape, a hexagon, a stylized letter "d", and a hexagon containing a lowercase "c". The word "INTERACTIVE" is written in a bold, white, sans-serif font.

®

PEER-TO-PEER NUMERIC COMPUTING WITH JAVASCRIPT

 Athan Reines /  @kgryte

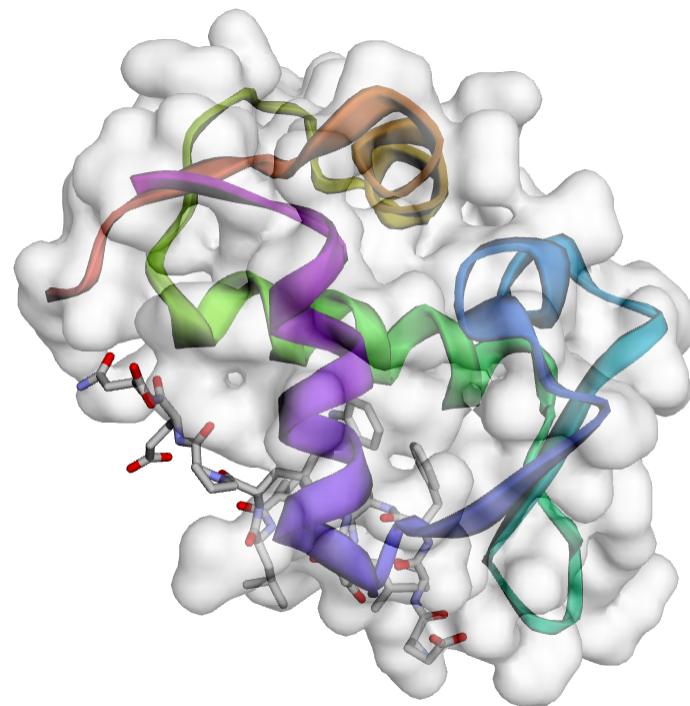
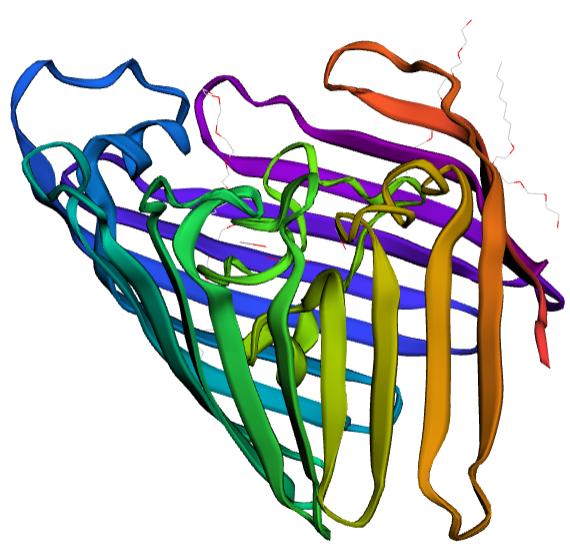
BLOOM FILTERS

DBSCAN

2D CLASSIFICATION

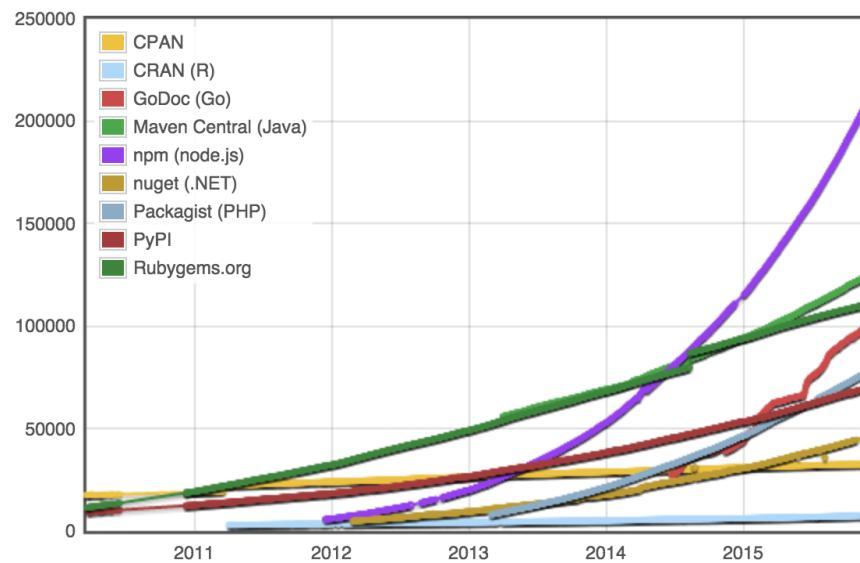
FOURIER SERIES

NEURAL NETWORKS



WHY JAVASCRIPT?

UBIQUITY



PERFORMANCE

	Fortran gcc 5.1.1	Julia 0.4.0	Python 3.4.3	R 3.2.2	Matlab R2015b	Octave 4.0.0	Mathematica 10.2.0	JavaScript V8 3.28.71.19	Go go1.5	LuaJIT gsl-shell 2.3.1	Java 1.8.0_45
fib	0.70	2.11	77.76	533.52	26.89	9324.35	118.53	3.36	1.86	1.71	1.21
parse_int	5.05	1.45	17.02	45.73	802.52	9581.44	15.02	6.06	1.20	5.77	3.35
quicksort	1.31	1.15	32.89	264.54	4.92	1866.01	43.23	2.70	1.29	2.03	2.60
mandel	0.81	0.79	15.32	53.16	7.58	451.81	5.13	0.66	1.11	0.67	1.35
pi_sum	1.00	1.00	21.99	9.56	1.00	299.31	1.69	1.01	1.00	1.00	1.00
rand_mat_stat	1.45	1.66	17.93	14.56	14.52	30.93	5.95	2.30	2.96	3.27	3.92
rand_mat_mul	3.48	1.02	1.14	1.57	1.12	1.12	1.30	15.07	1.42	1.16	2.36

Figure: benchmark times relative to C (smaller is better, C performance = 1.0).

IN-BROWSER ANALYSIS

DATA PIPELINES

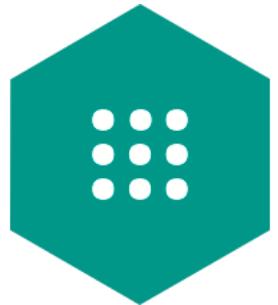
STREAMS

```
source.pipe( transform ).pipe( transform ).pipe( destination );
```

CLI

```
$ cat ./data.csv | node ./bin/filter | node ./bin/stats > ./out.txt
```

WHAT CAN YOU USE TODAY?



Data Structures

- > arrays
- > matrices
- > ndarrays (in progress)
- > data frames (in progress)

```
var matrix = require( 'dstructs-matrix' );

var mat = matrix( [5,2], 'int16' );
/*
[ 0 0
  0 0
  0 0
  0 0
  0 0 ]
*/

mat.sset( '1:3,:', 5 );
/*
[ 0 0
  5 5
  5 5
  0 0
  0 0 ]
*/
```

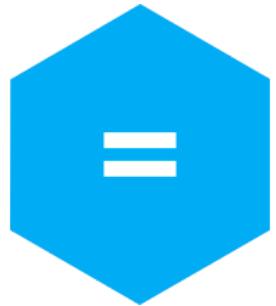


Validation

- > positive zero
- > safe integer
- > permutation
- > positive integer array
- > ...many others

```
var isUint32Array = require( 'validate.io-uint32array' );
var isPosIntArray = require( 'validate.io-positive-integer-array' );

function foo( x ) {
  if ( !isUint32Array( x ) && !isPosIntArray( x ) ) {
    throw new TypeError( 'invalid input argument. Value: `'+x+'`.' );
  }
  ...
}
```



Computation

- > error function
- > quantiles
- > fliplr
- > cosine similarity
- > ...many others

```
var matrix = require( 'dstructs-matrix' );
var mean = require( 'compute-mean' );

var mat = matrix( [7,3,9,11], [2,2], 'float64' );
/*
  [
    [ 7   3
      9  11 ]
*/
// Compute the mean across the columns:
var mu = mean( mat, { 'dim': 2} );
/*
  [
    [ 5
      10 ]
*/
```



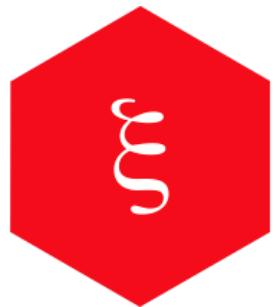
Distributions

- > probability density functions
- > moment generating functions
- > random variates
- > quantiles
- > ...many others

```
var randn = require( 'distributions-normal-random' );

// Seed the generator:
randn.seed = 52;

// Generate a matrix of random variates:
var mat = randn( [3,2], { 'dtype': 'float64' } );
/*
  [ -0.482  0.274
    0.725   1.113
    0.608   1.050 ]
*/
```

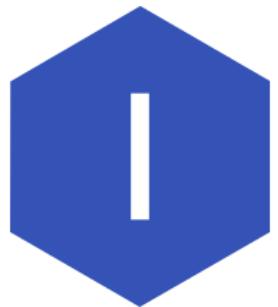


Random

- > lcg
- > random data
- > ...many others

```
var randc = require( 'rand-color-hexadecimal' ) ;

var hex = randc();
// returns '<color>'; e.g., '474747'
```



Streams

- > split
- > map
- > join
- > statistics
- > ...many others

```
var splitStream = require( 'flow-split' );
var mapStream = require( 'flow-map' );
var joinStream = require( 'flow-join' );

function map( value, idx ) {
  value = parseFloat( value );
  return (value * idx).toString();
}

var sStream = splitStream( { 'sep': '/\r?\n/' } );
var jStream = joinStream( { 'sep': '\n' } );
var mStream = mapStream( map );

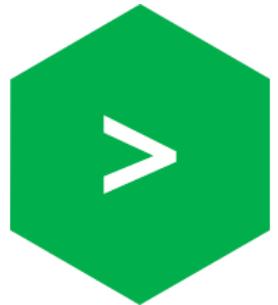
process.stdin
  .pipe( sStream )
  .pipe( mStream )
  .pipe( jStream )
  .pipe( process.stdout );
```

```
$ echo -n $'1\n2\n3\n4\n' | flow-map ./transform.js | <stdin>
```



Charts

- > timeseries
- > scatterplot
- > network
- > matrix diagrams
- > ...many others



Data

- > Iris
- > sentiment analysis
- > first name frequencies
- > ...many others

```
var toMatrix = require( 'dstructs-to-matrix' );
var iris = require( 'datasets-iris' );

var mat = toMatrix([
  iris.setosa.sepal.len,
  iris.setosa.sepal.width,
  iris.setosa.petal.len,
  iris.setosa.petal.width,
  iris.versicolor.sepal.len,
  iris.versicolor.sepal.width,
  iris.versicolor.petal.len,
  iris.versicolor.petal.width,
  iris.virginica.sepal.len,
  iris.virginica.sepal.width,
  iris.virginica.petal.len,
  iris.virginica.petal.width
]);

```



Notebook

- > interactive
- > preloaded
- > ...work in progress

WHERE ARE WE NOW?

- > > 1500 repos
- > > 1000 are public
- > >600 published modules
- > ...a lot more to come

FUTURE WORK

- > workflow
- > modules
- > documentation
- > community

`github.com/<org>/discussions`

PEER-TO-PEER

- wrtc
- simple-peer
- webrtc-connect
- multiplex
- rpc-multistream

SIMPLE

```
var server = rpc( methods );
var client = rpc();

client.pipe( server ).pipe( client );

client.on( 'methods', onMethods );

function onMethods( methods ) {
  methods.matrix( [0,1,2,3,4,5,6,7,8,9], [5,2], 'int16', onMatrix );
}

function onMatrix( err, matrix ) {
  if ( err ) {
    throw err;
  }
  ...
}
```

TCP

Server

```
var rpc = require( 'rpc-multistream' );
var net = require( 'net' );

var server = rpc( methods );
server.on( 'methods', runAnalysis );

net.createServer( onConnection ).listen( 4242 );

function onConnection( connection ) {
  connection.pipe( server ).pipe( connection );
}
```

TCP

Client

```
var rpc = require( 'rpc-multistream' );
var net = require( 'net' );

var client = rpc( methods );
client.on( 'methods', runAnalysis );

var connection = net.connect( { 'port':4242}, onConnect );

function onConnect() {
  connection.pipe( client ).pipe( connection );
}
```

WEBRTC

Peer 1

```
var rpc = require( 'rpc-multistream' );
var rtcc = require( 'webrtc-connect' );

var server = rpc( methods );
server.on( 'methods', runAnalysis );

rtcc.createServer( onPeer ).listen( 9999, '127.0.0.1' );

function onPeer( error, peer ) {
  if ( error ) {
    throw error;
  }
  peer.pipe( server ).pipe( peer );
}
```

WEBRTC

Peer 2

```
var rpc = require( 'rpc-multistream' );
var rtcc = require( 'webrtc-connect' );

var client = rpc( methods );
client.on( 'methods', runAnalysis );

rtcc.connect( { 'port':9999, 'url':'http://127.0.0.1' }, onPeer );

function onPeer( error, peer ) {
  if ( error ) {
    throw error;
  }
  peer.pipe( client ).pipe( peer );
}
```

DEMO

github.com/kgryte/talks-nodejs-interactive-2015

CONTRIBUTORS



⌚ Philipp Burckhardt
🐦 @burckhap



⌚ Robert Gislason



⌚ Rebekah Smith
🐦 @froodette

-  [github/kgryte](https://github.com/kgryte)
-  [npmjs/~kgryte](https://npmjs.com/~kgryte)
-  [@kgryte](https://twitter.com/kgryte)
-  kgryte@gmail.com

APPENDIX



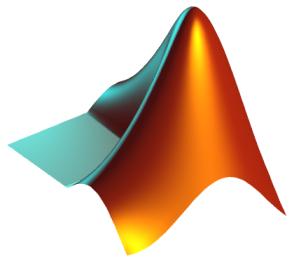
BIO



nodeprime

vurb

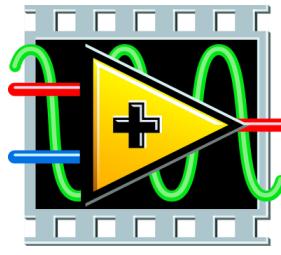
CONTEXT



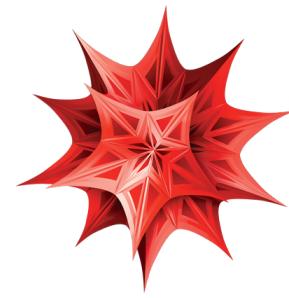
1970s



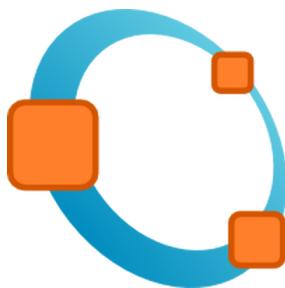
1980



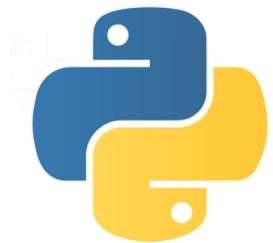
1985



1986



1988



1995



1997



2009



2005



2010



2011



2012



2005



2009



2011



2011

- > open source
- > web technologies
- > decoupling computation and consumption

WHY JAVASCRIPT?

VISUALIZATION

JSON

```
{  
    "type": "Matrix",  
    "dtype": "int8",  
    "shape": [5,2],  
    "offset": 0,  
    "strides": [2,1],  
    "raw": true,  
    "data": [4,2,13,1,1,8,21,9,9,11]  
}
```

WEB TECHNOLOGIES

ELECTRON



WHAT COULD BE BETTER?

- Int64 (and bitwise ops)
- Typed Objects
- WebCL
- SIMD (long)
- Parallel Computing
- Operator Overloading
- Web Assembly

INTEGER SUPPORT

- discussion
- gist
- Int64 in R

TYPED OBJECTS

- spec
- explainer
- typed data structures in Go

WEBCL

- node-opencl

SIMD

- polyfill
- Intel announcement
- MDN
- presentation

PARALLEL COMPUTING

- Data parallelism
- Task parallelism
- Scheduler
- Lock-free programming
- Shared Memory Web Workers

OPERATOR OVERLOADING

- operator-overloading-js
- paper.js
- paper.js source

THE END