Julia for big data
What we want
What we want

- data source
- data source
- data source

something fast and accurate
sometimes we’re not sure
often we want to do lots of things in parallel
making use of multiple cores, chips, memories, filesystems

insight
What we have to do

data source

cleanup
analyze
summarize
visualize
insight
sanity checks
What we have to do

- data source
- cleanup
- analyze
- summarize
- visualize
- sanity checks
- insight

reimplement for production
lock in
How can programming language design help with big data?

<table>
<thead>
<tr>
<th>Work with multiple data sources</th>
<th>API for large scale data, distributed, data source objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create multiple visualizations</td>
<td>API for visualization objects</td>
</tr>
<tr>
<td>Iterate through many different workflows</td>
<td>High level language constructs for rapid development</td>
</tr>
<tr>
<td>Interoperate with existing tools/libraries</td>
<td>Easy foreign function interface: C, Fortran, LLVM,…</td>
</tr>
<tr>
<td>Reproducible science automatable analysis</td>
<td>Consistent cross-platform interop</td>
</tr>
<tr>
<td>Fast throughput</td>
<td>Native language constructs for multithreading, parallel computing</td>
</tr>
</tbody>
</table>
Julia today

core functions
- parallel computing
- linear algebra
- datetime

package ecosystem
- FFT
- parallel computing
- code generation
- distributed arrays
- async I/O

external libraries
- CUDA
- OpenCL
- CuBLAS

other packages
- DataFrames
- other organizations
- other packages

IPython Notebook
- PETSc
- PETSc.jl
- BLAS
- LAPACK
- MKL
- Images.jl
- GSL.jl
- IJulia.jl

D3
- PyCall.jl
- Gadfly.jl

JuliaGPU
- JuliaOpt
- CVX.jl

JuliaStats
- JuliaOpt
- JuliaStats

ODBC.jl
- TimeSeries.jl

basic webstack
- Python
- ODBC

ecosystem
- D3
- Gadfly.jl
- PyCall.jl
- ODBC.jl

Unifying in-core and out-of-core algorithms

DataFrames

<table>
<thead>
<tr>
<th>Name</th>
<th>Height</th>
<th>Weight</th>
<th>Gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>73.0</td>
<td>NA</td>
<td>Male</td>
</tr>
<tr>
<td>Jane Doe</td>
<td>68.0</td>
<td>130</td>
<td>Female</td>
</tr>
</tbody>
</table>

file I/O

DataFrames.jl

julia code

HDF5.jl

Myria

Airline data = SQL

CREATE VIEW VW_AIRLINE_TIMES_UTC AS
SELECT ROW_NUMBER() OVER (ORDER BY FLIGHT TAKEOFF_DATETIME_ORIGIN) AS UNIQUE_FLIGHT_NUMBER,
ORIGIN, FLIGHT TAKEOFF_DATETIME_ORIGIN,
FLIGHT_LANDING_DATETIME_ORIGIN, TIMEZONE FROM AIRLINE
LEFT JOIN AIRPORT ON (AIRLINE.ORIGIN = AIRPORT.IATA_CODE);

SQL or ODBC-aware database

Multiple dispatch + metaprogramming (code generation)
Linear algebra

Today:

• Wrappers to OpenBLAS, LAPACK, SuiteSparse, ARPACK

• Generic routines for LU, Cholesky, QR, etc. over arbitrary fields (rationals, quaternions, arbitrary precision arithmetic, etc.)

Planned:

• new abstractions for iterative methods [IterativeSolvers.jl#28, #29]

• randomized algorithms efficient on huge data sets: for least squares, SVD,…

• Native scalable solvers that are faster than LAPACK in serial, yet parallelizable to arbitrarily many cores

• record breaking sparse matrix solvers for size and speed
Community-driven statistics and machine learning in Julia

### JuliaStats

**Statistics and Machine Learning** made easy in Julia.

- Easy to use tools for statistics and machine learning.
- Extensible and reusable models and algorithms.
- Efficient and scalable implementation.
- Community driven, and open source.

### Packages

#### MLBase
Swiss knife for machine learning
- Data preprocessing
- Score-based classification
- Performance evaluation
- Model selection, cross validation

#### Distance
Various distances between vectors
- A large variety of metrics
- Efficient column-wise and pairwise computation
- Support weighted distances

#### KernelDensity
Kernel density estimation
- Kernel density estimation for univariate and bivariate data
- User customization of interpolation points, kernel, and bandwidth

#### StatsBase
Basic functionalities for statistics
- Descriptive statistics and moments
- Sampling with/without replacement
- Counting and ranking
- Autocorrelation and cross-correlation
- Weighted statistics

#### DataArrays
Arrays that allow missing data
- Data arrays with missing values
- Optimized representation of arrays comprised of repetitive values
- Computational routines that work with missing values

#### DataFrames
Essential tools for tabular data
- DataFrames to represent tabular datasets
- Database-style joins and indexing
- Split-apply-combine operations, pivoting
- Formula and model frames

#### Clustering
Algorithms for data clustering
- K-means
- K-medoids
- Affinity propagation
- Evaluation of clustering performance

#### GLM
Generalized linear models
- Friendly API for fitting GLM to data
- Work with data frames and formulas
- A variety of link types
- Optimized implementation

#### NMF
Nonnegative matrix factorization
- A variety of NMF algorithms, including Lee & Seung’s, Projected ALS and projected gradient, with optimized implementation.
- NDNDDO initialization

#### Distributions
Probability distributions
- A large collection of univariate, multivariate distributions
- Descriptive stats, pdf/pmf, and rnf
- Efficient sampling
- Maximum likelihood estimation

#### MultivariateStats
Multivariate statistical analysis
- Linear regression (LSQ and Ridge)
- Dimensionality reduction (PCA,ICA,CA...)
- Multidimensional scaling
- Linear discriminant analysis

#### HypothesisTests
Hypothesis tests
- Parametric tests: t-tests
- Nonparametric tests: binomial tests, sign tests, exact tests, U tests, rank tests, etc

#### RegERMs
Regularized empirical risk minimization
- Foundational framework for regression analysis
- Support ridge regression, logistic regression, and many more (e.g. user-provided loss)
- Solvers: L-BFGS and SGD
- Highly configurable and

#### MCMC
Markov Chain Monte Carlo
- A generic engine for Bayesian inference
- A variety of samplers, using latest techniques
- User-friendly syntax for model specification
- Use auto-differentiation
- Ability to suspend and resume

#### TimeSeries
Time series analysis
- Tools to represent, manipulate, and apply computation to time series data
- Based on data frames
Hardware accelerators and Julia

Today:

- **JuliaGPU** organization for GPU-based computing
- Wrappers for CUDA [CUDA.jl] and OpenCL [OpenCL.jl]

Planned and in progress:

- LLVM code generation targeting Intel Xeon Phi (KNC, KNL architectures)
- Unified model for heterogeneous hardware threads
Some recent big data-related Julia improvements

<table>
<thead>
<tr>
<th>Better I/O</th>
<th>&gt;20x faster HDF5 I/O [HDF5.jl#132]</th>
</tr>
</thead>
</table>
| Better data handling             | comprehensive handling of dates and times [#7654]  
array views [#7941] and Cartesian iteration [#6437]  
string handling                   |
| Better parallel computing        | faster interprocess communication with 10x faster serialization [#7893] 
multithreading support [#1790, #6741, threads] |
| Better foreign code interop      | llvmcall [#5046], C++ [CXX.jl], Java [JavaCall.jl] 
staged functions [#7474] 
C struct interoperability [#7906] |
| Interactive visualization support| reactive programming framework [React.jl] 
interactive IPython widgets [IJuliaWidgets.jl] |
| Other speedups                   | better garbage collection [#5227] |

...and many more planned [0.4-projects]
Stretch goal: To break the record on the largest data set that's ever been analyzed.
The MIMIC II dataset
(with S. Madden & group)

raw waveform data

SQL database

mimic.physionet.org
The MIMIC II dataset

(with S. Madden & group)

Real-world problems

- raw waveform data
- files in custom binary data format
- limited data for a few patients
- SQL database

missing data

- corrupt data (purposely anonymized)
- free-form data (doctors’ notes, prescriptions, …)

mimic.physionet.org
Topological data analysis

raw data | persistent homology | visualizations
From paper to Julia code

**Definition 2.1.** An abstract simplicial complex is a pair \((V, \Sigma)\), where \(V\) is a finite set, and \(\Sigma\) is a family of non-empty subsets of \(V\) such that \(\sigma \in \Sigma\) and \(\tau \subseteq \sigma\) implies that \(\tau \in \Sigma\). (G. Carlsson, *Bull. Amer. Math. Soc.* 46 (2009), 255-308.)

```julia
In []:

type AbstractSimplicialComplex{T} =
    V :: Vector{T}  #Vertex set with total ordering imposed
    Σ :: Set{Vector{T}}  #Set of Vs

function AbstractSimplicialComplex{T}(V::Vector{T}, Σ)
    Σ' = Set{Vector{T}}()
    for σ in Σ
        σ ⊆ V || error("σ=$σ ∈ Σ is not a subset of V")
        push!(Σ', σ)
    end
    #Check that σ∈Σ and τ∈σ implies that τ∈Σ
    for σ in Σ, τ in ϕ(σ)
        τ ∈ Σ || error("τ=$τ ⊆ σ=$σ but τ∉Σ")
    end
    new(V, Σ')
end
```